REPORT DOCUMENTATION PAGE

AFRL-SR-BL-TR-00-

0418

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	...COVERED (From - To)
17-07-2000	Final Technical Report	01/06/1997 - 31/05/2000

4. TITLE AND SUBTITLE
AASERT 97 Constrained Control Allocation Methods for Reconfigurable Flight Control Laws

5a. CONTRACT NUMBER

5b. GRANT NUMBER
F49620-97-1-0405

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Marc Bodson

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Department of Electrical Engineering
University of Utah
50 S Central Campus Dr Rm 3280
Salt Lake City, UT 84112-9206

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Dr Marc Jacobs
AFOSR/NM
801 North Randolph Street, Room 732
Arlington, VA 22203-1977

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes.

13. SUPPLEMENTARY NOTES
The views and conclusions contained in the report are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government.

14. ABSTRACT

The objective of the project was to develop and analyze methods for control in the presence of actuator saturation. Algorithms were investigated for control allocation, that is, for the problem of distributing control requirements among redundant actuators. A new implementation of the direct allocation method of Durham was proposed. The direct allocation method was chosen because it utilized all of the attainable moment set. A special representation of the moment set in spherical coordinates was considered to speed-up the real-time computations and two rapid search methods were developed and successfully implemented. The direct allocation method was also extended to systems that had previously been excluded, namely those for which subsets of three actuator commands produce linearly dependent moments. Testing of the algorithms was performed using simulation models of a military transport aircraft and of a tailless aircraft.

15. SUBJECT TERMS
Flight control systems, control allocation, multivariable systems, constrained dynamical systems, reconfigurable control.

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Professor Marc Bodson
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL	49	19b. TELEPHONE NUMBER (include area code) (801) 581 8590

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

Final Technical Report: AASERT 97
Constrained Control Allocation Methods
for Reconfigurable Flight Control Laws

AFOSR F49620-97-1-0405

Marc Bodson
Department of Electrical Engineering
University of Utah

Objectives

The surfaces that control aircraft trajectories have a limited range and rate of motion. The limits can lead to a significant degradation of performance if not accounted for in the design process. Instability may even occur. The objective of the project was to develop and analyze methods for control in the presence of actuator saturation. Algorithms were investigated for control allocation, that is, for the problem of distributing control requirements among multiple actuators when redundancy is available. Of particular interest were methods that exploited all of the available control power and that were implementable in real-time. Such objectives are important for flight control systems that are designed to reconfigure automatically after failures and damages, because of the loss of control authority. Testing of the algorithms was performed using simulation models of a military transport aircraft and of a tailless aircraft.

Summary of the Results

A new control allocation method was developed, based on the direct allocation method of Durham. The direct allocation method was chosen because it utilized all of the attainable moment set. The focus of the investigations was on developing techniques that would speed-up the real-time computations. A special representation of the moment set in spherical coordinates was considered and two rapid search methods were developed and successfully implemented. The direct allocation method was also extended to a class of systems that had previously been excluded, namely systems for which subsets of three actuator commands produce linearly dependent moments .

Two US graduate students (one Ph.D. and one M.S.) were supported by the project. Two undergraduate students were also supervised on a B.S. project, and started graduate study afterwards. The research was performed in conjunction with the AFOSR awards: "Robust adaptive algorithms for reconfigurable flight control" (Grant: F49620-95-1-0341, the original parent grant) and "Self-designing control systems for piloted and uninhabited aerial vehicles" (Grant: F49620-98-1-0013).

Accomplishments

Control Allocation

Control allocation is the problem of distributing control requirements among redundant control surfaces. The problem is particularly important for tailless aircrafts (because current designs involve a large number of control surfaces) and for reconfigurable control laws (because control surfaces need to be commanded separately for the maximum capabilities of the aircraft to be exploited). The following state-space model was considered

$$\dot{x} = Ax + Bu + d$$
$$y = Cx$$

where $x \in \mathrm{R}^n$, $d \in \mathrm{R}^n$, $u \in \mathrm{R}^m$, $y \in \mathrm{R}^p$. For the control of aircrafts, the states given by the vector x may include the angle of attack, the pitch rate, the angle of sideslip, the roll rate, and the yaw rate ($n=5$). The output vector y may contain the pitch rate, the roll rate, and the yaw rate ($p=3$). The control input vector u consists in the control surface deflections, or in the commanded actuator positions if the actuator dynamics can be neglected. If the control variables are ganged, m may be as small as 3. Otherwise, the typical range is $m=5 \rightarrow 20$.

Model reference control laws, sometimes referred to as dynamic inversion control laws, rely on a reference model which represents the desired dynamics of the closed-loop system, for example

$$\dot{y}_M = A_M y_M + B_M r_M$$

where r_M is a reference input vector, and y_M represents the desired output of the system. Since the derivative of y is given by

$$\dot{y} = CAx + CBu + Cd$$

the objective may be achieved by setting

$$u = (CB)^{-1}\left(-CAx - Cd + A_M y + B_M r_M\right)$$

Model matching follows if the matrix CB is square and invertible, and if the original system is minimum phase. Adaptive implementations of this control law were discussed in the context of reconfigurable flight control in [1].

If the matrix CB is not full rank, model matching may still be possible, but with a different model and a more complex control law. On the other hand, if CB is not square but full row rank (more columns than rows, as in the case of redundant actuators), the same model reference control law can be used if one defines

$$a_d = -CAx - Cd + A_M y + B_M r_M$$

and if the control input u is such that

$$(CB)u = a_d$$

Obtaining u requires that one solve a system of linear equations with more unknowns than equations. This may seem like an easy problem, but the difficulty in practice is that the vector u is constrained. The limits generally have the form

$$u_{\min,i} \le u_i \le u_{\max,i} \qquad \text{for } i = 1,...,m$$

or, $u_{\min} \le u \le u_{\max}$, in vector form. Given these limits, an exact solution may not exist, despite the redundancy.

The control allocation problem generally consists in finding the control input u that best fits the desired relationship, while satisfying the constraints. Towards that objective, the *direct allocation method* of Durham ("Constrained Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 16, no. 4, 1993, pp. 717-725) defines the following problem: given a desired vector m_d, find the vector u such that CBu is closest to m_d in magnitude, with u satisfying the constraints and CBu proportional to m_d. In the Durham's formulation, the vector m_d was a desired moment.

At the core of Durham's method is the computation of the set of attainable moments (or set of attainable accelerations here, a minor adjustment). Fig. 1 shows the sets of attainable accelerations for a C-17 transport aircraft model (on the left) and for a tailless fighter aircraft model (on the right). The axes of the plots are the pitch, roll, and yaw accelerations in deg/s^2. The boundaries of the sets specify the maximum accelerations obtainable under the control limits, assuming linear models for the aircraft dynamics. What makes the control allocation problem difficult is the large number of actuators: 11 for the tailless aircraft and 16 for the C-17 aircraft. A computer code was developed that produces the sets shown on Fig. 1 in a fraction of a second.

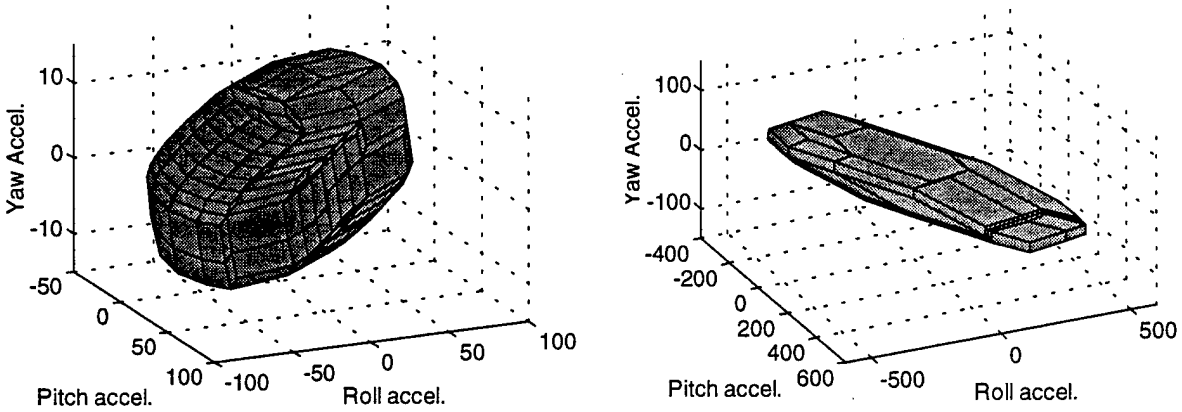


Fig. 1: Sets of attainable accelerations,
C-17 aircraft (left) and tailless aircraft (right)

The set for the tailless aircraft is delimited by 78 facets, while the set for the C-17 is determined by 240 facets. The method developed by Durham and co-workers requires

that, if one extracts any three columns of the *CB* matrix, the resulting matrix is nonsingular. This condition is satisfied for the C-17 model used for Fig. 1 (on the left) and implies that every facet is a parallelogram. For the tailless model, the condition is not satisfied. In fact, some columns of the matrix are even linearly dependent (because pitch thrust vectoring and pitch flaps only produce pitching accelerations). To obtain the plot shown on the right of Fig. 2, the original method was extended to relax the linear independence requirement. Note two polygonal facets visible on the plot.

The most significant part of the computations in Durham's method is performed to obtain the set of attainable accelerations. To compute the control signals after the set is obtained, one must determine the facet towards which the desired acceleration points, and then perform some simple computations. Since the determination of the set may be performed off-line, the method becomes a fast algorithm for control allocation guaranteeing the use of the maximum control authority, *if the applicable facet can be found rapidly.*

The use of spherical coordinates was investigated as a way to perform the search. The results of this study are reported in [2], [3], and [4]. On Fig. 2 is a representation of the two sets of Fig. 1 in spherical coordinates. Note that, because the determination of the applicable facet is only dependent on the direction of the desired acceleration, the search problem is effectively a 2-dimensional problem.

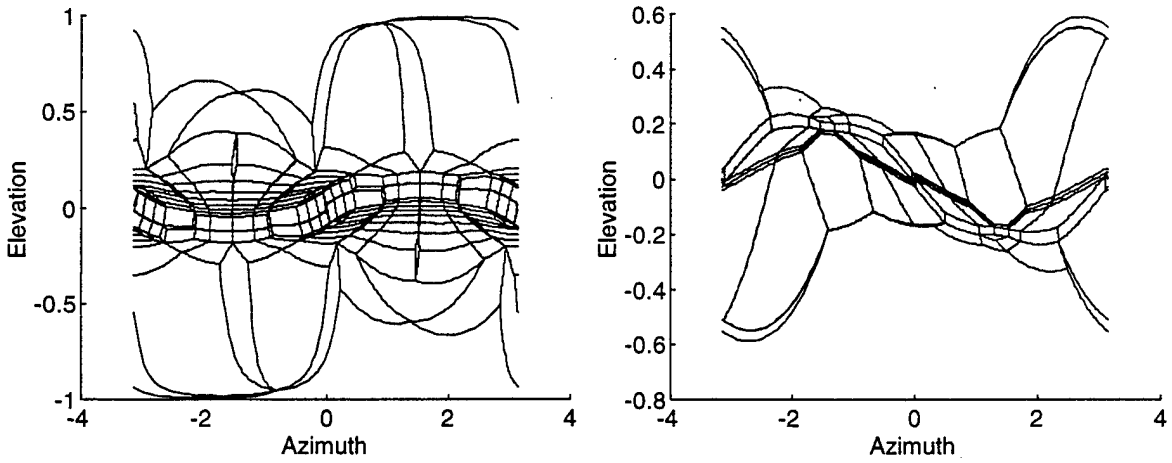


Fig. 2: Sets of attainable accelerations expressed in spherical coordinates, C-17 aircraft (left) and tailless aircraft (right)

Two methods were developed using the idea of spherical coordinates. In the first method, ranges were pre-computed for the coordinates of the facets, defining boxes that contained the facets. The search was similar to an exhaustive search of the facets, except that simple inequality tests were used to quickly eliminate facets from the search. For those facets that satisfied the box check, a more complicated test was performed. This test

conclusively established whether the facet was the correct one. If the test was successful, the control input was rapidly obtained. The idea was that the more complicated test was only required for very few facets. In experiments with the C-17 model, the test was required once in 50% of the cases and less than three times in 95% of the cases. Overall, it was required an average of 1.97 times, as opposed to 49.7 times in the case of an exhaustive search.

A second method was also developed using the concept of spherical coordinates. In that method, a table of facets was created off-line, and the table was indexed by spherical coordinates. The determination of the applicable facet was then achieved by simple table look-up. This option required virtually no on-line computations and provided a guaranteed solution in a fixed time. Its drawback was a potentially large memory requirement and longer off-line execution times.

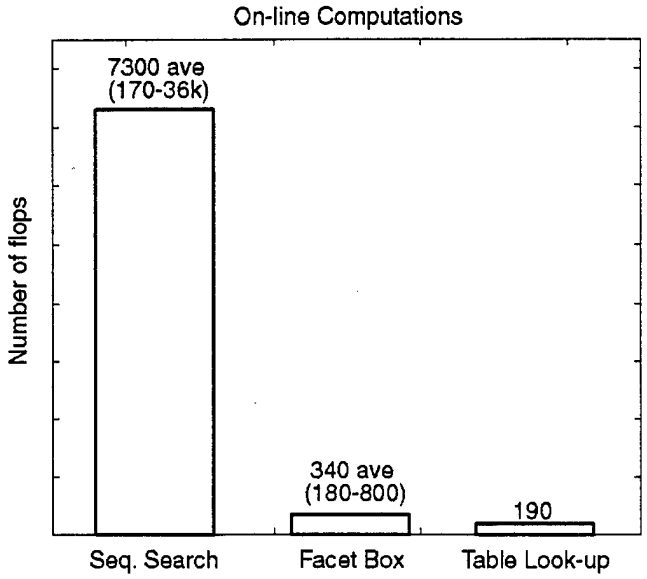


Fig. 3: Comparison of the on-line computations required by the different methods (C-17 aircraft)

Fig. 3 gives a comparison of the on-line computations required by the two methods, together with a baseline provided by a sequential search of the facets. The table entries for the sequential search and facet box methods have a range and an average value. The range denotes the minimum to maximum (worst-case) possible values for the method, whereas the average was computed for 1000 randomly selected moments. As mentioned earlier, the facet box approach considerably reduces the computations required, as compared to a sequential search. The table look-up approach goes even further and virtually eliminates the computations to be performed on-line, as well as the variability in the number of those computations. Memory requirements are significant, however.

Similar techniques were applied to systems which did not satisfy the linear independence assumption. As shown in Fig. 1, Durham's condition was relaxed for the determination of

the attainable acceleration set by replacing the parallelogram facets by polygons. To perform control allocation, two options were considered. The first option used representations of the polygonal facets in spherical coordinates. It turned out to be difficult to implement and was not further pursued. A second option was explored that consisted in representing the polygonal facets as superimposed parallelogram facets. This option had the advantage of sharing many features with the original method. Given a desired acceleration, the table look-up method produced several sub-facets, each of them yielding a valid control input (the solution may not be unique if the linear independence assumption is not satisfied). In order to insure the continuity of the solution, the solution that was picked was the average of the solutions corresponding to each sub-facet.

Fig. 4 shows a block diagram representation of the table look-up implementation of Durham's method, as proposed in [2], [3], and [4]. The resulting algorithm has the advantage of being feasible in real-time and to guarantee usage of the maximum control authority. Its reliability is also excellent, as the computations are very simple and their results predictable. The main drawback of the method is that it requires a substantial memory space. Some computations must be performed off-line for the creation of the look-up tables and, in a reconfigurable control law, they must be performed on-line. However, they may be carried out at a lower computational rate, that is, the rate associated with the variation of the adaptive parameters.

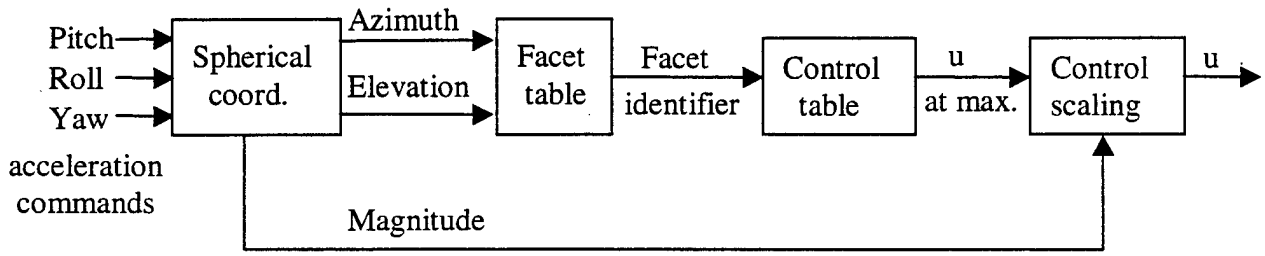


Fig. 4: Block diagram of the table look-up algorithm
for direct allocation using spherical coordinates

Flight Testing with Remote-Controlled Aircraft

A small flight testing platform was developed using commercial R/C aircraft hardware. The objective of this effort was primarily educational, and a tool to attract students to the field of intelligent aircraft systems. Two undergraduate students completed senior projects on this topic. The nose cone developed by the students is visible on the photo of the aircraft on Fig. 5, and provides air pressure, angle of attack and angle of sideslip measurements. As opposed to other similar projects, the airframe is based on a commercial almost-ready-to-fly (ARF) R/C aircraft and low-cost instrumentation (no GPS or sophisticated on-board computer), with the objective of performing experiments which would not be considered feasible with conventional flight testing platforms.

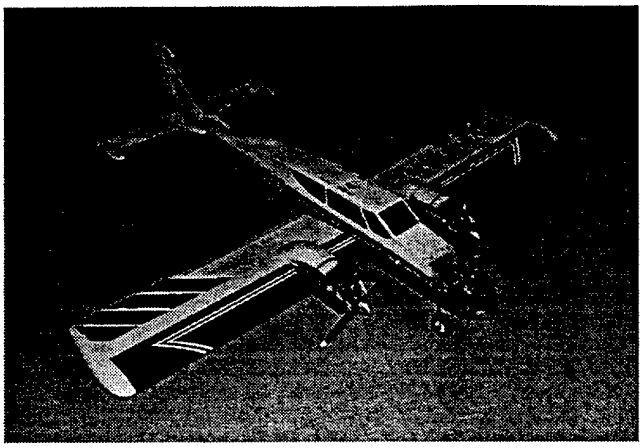


Fig. 5: Remote-controlled aircraft developed for flight control experiments

Personnel Supported

Graduate Students: Mark Leatherwood and John Petersen, both US citizens, were supported by the grant. David Shore and Dan Stevens, also both US citizens, completed their B.S. project under the PI's supervision. They did not receive any salary, but some funds were provided for the development of their experiments.

Interactions/Transitions

John Petersen presented a paper at the *AIAA Guidance, Navigation, and Control Conference* (Portland, OR, August 9-11, 1999), describing the results of his work.

Publications

[1] M. Bodson & W. Pohlchuck, "Command Limiting in Reconfigurable Flight Control," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 21, no. 4, pp. 639-646, 1998.

[2] J. Petersen & M. Bodson, "Fast Control Allocation Using Spherical Coordinates," *Proc. of the AIAA Guidance, Navigation, and Control Conference*, Portland, OR, pp. 1321-1330, 1999.

[3] J. Petersen & M. Bodson, "Control Allocation for Systems with Coplanar Controls," to appear in the *Proc. of the AIAA Guidance, Navigation, and Control Conference*, Denver, CO, August 2000.

[4] J. Petersen & M. Bodson, "Fast Implementation of Direct Allocation and Extension to Coplanar Controls," submitted for publication to the *AIAA Journal of Guidance, Control, and Dynamics*, 2000.

Progress of the Students Associated with this Project

John Petersen and Mark Leatherwood made excellent progress towards their graduate degree with the support of the grant. John Petersen passed his Ph.D. qualifying examination and is expected to defend his proposal for Ph.D. dissertation in August 2000. Mark Leatherwood defended his M.S. thesis proposal in May 2000, and is expected to complete his thesis in August 2000. Dan Stevens was admitted for graduate study at U.C. Berkeley and moved there after the completion of his B.S. at the University of Utah. David Shore joined the graduate program of the Department of Mechanical Engineering at the University of Utah, and will pursue an M.S. degree under the supervision of the PI of this grant. For his thesis, he will continue the work of his B.S. project and implement real-time identification algorithms for air vehicles on a remotely-piloted aircraft.

JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 21, No. 4, July–August 1998

Command Limiting in Reconfigurable Flight Control

Marc Bodson* and William A. Pohlchuck†
University of Utah, Salt Lake City, Utah 84112

Limits on the motion and on the rate of motion of the actuators driving the control surfaces of aircraft significantly affect the performance of flight control systems. After a failure or damage to the aircraft, the constraints become even more restrictive because of the loss of control power. There is also often an increase in cross couplings between the axes and, for a period of time, a significant uncertainty about the moments generated by the individual control surfaces. A model reference adaptive control algorithm is considered for flight control reconfiguration. The tracking performance of the algorithm deteriorates drastically for large maneuvers if actuator saturation is not accounted for. Four methods of command limiting are proposed to handle the problem, which are based on a scaling of the control inputs, a relaxation of the control requirements, a scaling of the reference inputs, and a least-squares approximation of the commanded accelerations. Simulations demonstrate the effectiveness of the algorithms in the reconfigurable flight control application. Even the simplest method is found to considerably improve the responses, and, surprisingly, the performance of all four methods is similar despite their widely different concepts and complexity levels. In some cases, degraded transient responses are observed, which are attributed to the uncertainty in the aircraft parameters following a failure.

Introduction

ACTUATOR limits pose a major problem in flight control system design. The positions of control surfaces are limited both in their range and in their rate of motion. If not accounted for in the design, in flight these constraints may lead to a significant degradation of performance and, sometimes, pilot-induced oscillations. The origin of these problems is in the delay of the response of the aircraft resulting from actuator saturation, as well as in the multivariable nature of the control problem, which is such that saturation in one axis may lead to poor responses in other axes. Command limiting is the problem of modifying the actuator commands so that the control objectives are achieved in the best possible manner despite the control constraints. A more general problem is that of control allocation, which includes the requirement of distributing the control activity among multiple actuators, if redundancy is available.

For reconfigurable flight control systems, actuator saturation becomes even more problematic. The reason is that not only is control power reduced, but also that couplings between longitudinal and lateral axes are significantly larger after many failures. In addition, command limiting has to be performed in the presence of considerable uncertainty regarding the moments generated by the control inputs. Although the same techniques may be applied for reconfigurable control systems as for nonadaptive systems, certain differences are expected to arise. This paper focuses on those problems and on the design of command-limiting methods in conjunction with control reconfiguration.

Two main approaches may be distinguished for the design of control laws under constraints. The first consists in taking the constraints into account in the design of the control law. For example, a constrained-optimization procedure may be applied to calculate the values of the control inputs needed to optimize some cost criterion subject to the constraints. This approach is the most elegant, but requires a considerable amount of computations and is difficult to implement in real time for a multivariable system. A second approach consists in designing the control laws without accounting for the control limits and devising a separate procedure to modify the control signals when the constraints are not satisfied. For example, admissible control inputs may be calculated to approximate the desired control inputs in some optimal sense within the constraints. Or the control objectives may be relaxed until the constraints are satisfied. The second approach has the advantages that the nominal control system design is separate from the treatment of control saturation and that the resulting algorithm is of a complexity amenable to real-time implementation.

Previous work includes the concept[1–3] of finding an allowable control input such that the generated moment in the direction of the desired moment is closest to the desired value. Another idea[4] is to find an approximation of the acceleration that is the closest to the desired value in a least-squares sense. In that case, the direction of the desired value does not take a particular role. Yet another approach[5] is to modify the reference input so that the approximate reference input applied to the control law does not yield saturation. In this paper, we study four methods of command limiting called scaling of control inputs, relaxation of control requirements, scaling of reference inputs, and least-squares approximation of commanded accelerations. Although the concepts have their roots in earlier papers, a novelty of this paper is the comparison of the methods and their evaluation in the context of reconfigurable flight control systems. The respective advantages of the methods are discussed in terms of computations, ease of implementation, and performance in simulations using a detailed fighter aircraft model.

Problem Statement

Aircraft Model

Consider the linearized aircraft model

$$\dot{x} = Ax + Bu + d, \qquad y = Cx \tag{1}$$

where $x \in R^5$, $u \in R^3$, $d \in R^5$, and $y \in R^3$. The states of the aircraft are given by x and include angle of attack α, pitch rate q, sideslip β, roll rate p, and yaw rate r. The control inputs are given by u and include elevator command δ_E, aileron command δ_A, and rudder command δ_R. The vector d is equal to $-Ax^* - Bu^*$, where x^* and u^* are the trim state and trim input, respectively. In this formulation, the trim terms are grouped together as a constant disturbance applied to the system. It is assumed that the whole state x is available for measurement, although only the output y is to be tracked. The output y includes q, p, and r, a choice that is adequate for low dynamic pressure and limited angle of attack.[4]

Note that in the design of a reconfigurable flight control system, it may be advantageous to separate the control inputs associated to each of the elevators (or stabilators) and to each of the ailerons. Such choice gives more flexibility, in particular to generate rolling

Presented as Paper 97-3604 at the AIAA Guidance, Navigation, and Control Conference, New Orleans, LA, Aug. 11–13, 1997; received Aug. 24, 1997; revision received Feb. 19, 1998; accepted for publication Feb. 25, 1998.

*Associate Professor, Department of Electrical Engineering. Senior Member AIAA.

†Graduate Student, Department of Electrical Engineering; currently Senior Engineer, International Space Station Program, The Boeing Company, Houston, TX.

moments from the elevators. This approach, however, significantly increases the number of parameters that must be identified online. Commands to the individual control surfaces must also be linearly independent functions of time to guarantee identifiability, a condition that may be difficult to guarantee. For simplicity, it is assumed that elevators and ailerons are paired together, so that the number of inputs and outputs are equal.

Model Reference Control Law

To evaluate the performance of the command-limiting methods, a specific but relatively simple multivariable control law is considered. The control objective is the tracking of a reference model. The output y is expected to match the output y_M of

$$\dot{y}_M = -ky_M + kr_M \tag{2}$$

where $y_M \in R^3$ and $r_M \in R^3$ and $k > 0$ is a design parameter. The same parameter k is used for the three variables and, for the simulations, the constant k was set to 2.5 rad/s (Ref. 4). This is not a restriction of the algorithm, and different dynamics may be chosen for the three axes, if desired. The vector r_M is the reference input and represents pilot commands of pitch rate, roll rate, and yaw rate, respectively. For a solution to exist, it is assumed that $\det(\boldsymbol{CB}) \neq 0$ and that the plant transfer function is minimum phase. These assumptions were analyzed earlier[6] and were found to be acceptable for the problem under consideration.

The state feedback control law

$$u = C_0 r_M + G_0 x + v \tag{3}$$

is used, where $C_0 \in R^{3\times 3}$, $G_0 \in R^{3\times 5}$, and $v \in R^3$ are controller parameters. The closed-loop response, in terms of the output y, satisfies

$$\dot{y} = \boldsymbol{CA}x + \boldsymbol{CB}u + \boldsymbol{Cd} = (\boldsymbol{CA} + \boldsymbol{CB}G_0)x + \boldsymbol{CB}C_0 r_M + \boldsymbol{CB}v + \boldsymbol{Cd} \tag{4}$$

and leads to the same input/output relationship as that of the reference model (2) for the so-called nominal values of the controller parameters:

$$C_0^* = k(\boldsymbol{CB})^{-1}, \qquad G_0^* = (\boldsymbol{CB})^{-1}(-\boldsymbol{CA} - k\boldsymbol{C})$$
$$v^* = -(\boldsymbol{CB})^{-1}(\boldsymbol{Cd}^*) \tag{5}$$

For control reconfiguration, the problem is to design an adaptive algorithm that estimates the controller parameters (5).

Adaptation Algorithm

Equation (4) can be viewed as a linear equation relating unknown parameters $\boldsymbol{CB}$, $\boldsymbol{CA}$, and $\boldsymbol{Cd}$, to known signals $\dot{y}$, x, and u. A recursive least-squares algorithm with a forgetting factor was judged appropriate for the estimation, and a stabilized version of the algorithm[7,8] was chosen because it provided a stabilized covariance matrix update and a variable forgetting property, such that convergence was faster when more information was available. These properties are particularly useful for flight control reconfiguration.

The equations for the algorithm are as follows. Define C_2, G_2, and v_2 to be the estimates of $\boldsymbol{CB}$, $\boldsymbol{CA}$, and $\boldsymbol{Cd}$. Next, create the 9×3 matrix of parameter estimates

$$\theta = (C_2 \quad G_2 \quad v_2)^T \tag{6}$$

and the 9×1 regressor vector

$$w^T = (u^T \quad x^T \quad 1)^T \tag{7}$$

For the true parameter θ^*, one has that $\dot{y} = \theta^{*T} w$. The equations for the adaptive algorithm are

$$P^{-1}[n] = \lambda P^{-1}[n-1] + w[n]w^T[n] + \alpha(1-\lambda)I \tag{8}$$

with the initial condition $P^{-1}[0] = \alpha I$, and

$$\theta[n] = \theta[n-1] + P[n]w[n](\dot{y}^T[n] - w^T[n]\theta[n-1]) + \alpha\lambda P[n](\theta[n-1] - \theta[n-2]) \tag{9}$$

Two constants of the algorithm must be chosen so that $0 < \lambda < 1$ and $\alpha > 0$. The implementation of the algorithm requires the knowledge of $\dot{y}$, which may be obtained from accelerometer measurements or through filtered differentiation of rotational rate measurements. To avoid the computation of the inverse of the matrix P, an approximation of the algorithm was used.[8] Further, it was found useful to normalize the signals, so that both $\dot{y}$ and w were divided by $\sqrt{(1 + cw^T w)}$ before being applied to the algorithm, with $c > 0$ a design parameter.

Given C_2, G_2, and v_2, the estimates of $\boldsymbol{CB}$, $\boldsymbol{CA}$, and $\boldsymbol{Cd}$, a certainty equivalence control law calculates the controller parameters:

$$C_0^* = kC_2^{-1}, \qquad G_0^* = C_2^{-1}(-G_2 - k\boldsymbol{C}), \qquad v^* = -C_2^{-1}v_2 \tag{10}$$

In theory, a problem presents itself if C_2 is singular or close to singularity. Such a case did not arise in the simulations performed, but could be handled by freezing the controller parameter matrix C_0 over periods of time where any element of C_2^{-1} exceeds a specified bound. Previous experience with the control algorithm was reported in related references.[6,9]

Command Limiting Methods

Control Constraints

The control input u is assumed to be constrained so that u must belong to an admissible position control set $\mathcal{U}_p$, where $\mathcal{U}_p = \{u \mid \text{for } i = 1, \ldots, 3, p_{i,\min} \leq u_i \leq p_{i,\max}\}$. The rate of variation of u is also constrained so that $|\dot{u}_i| \leq d_{i,\max}$ for $i = 1, \ldots, 3$. According to the simulation model used, the rate limits are assumed to be symmetric. However, this assumption may be easily removed.

Because the application to digital flight control is considered, the constraint on the rate of variation can then be translated into a position constraint. Given a control input $u(n-1)$ at the previous time sample and a sampling period T, the admissible control rate set is defined to be $\mathcal{U}_r$, where $\mathcal{U}_r = \{u \mid \text{for } i = 1, \ldots, 3, u_i(n-1) - Td_{i,\max} \leq u_i(n) \leq u_i(n-1) + Td_{i,\max}\}$. Both the position and the rate limits can be translated into a single admissible control set, $\mathcal{U} = \mathcal{U}_p \cap \mathcal{U}_r$, with $\mathcal{U}$ of the form $\{u \mid \text{for } i = 1, \ldots, 3, u_{i,\min} \leq u_i \leq u_{i,\max}\}$.

Method 1: Scaling of Control Inputs

The first method is the simplest. The desired control input is denoted u_d and the control input produced by the command limiting method is denoted u. The idea behind the method is that when u_d does not belong to the admissible control set, its components are scaled until the constraints are satisfied. The objective is to preserve the directionality of the commands while satisfying the constraints. Specifically, algorithm 1 is defined as follows.

1) Let ρ_1 be the largest number such that $0 \leq \rho_1 \leq 1$ and $u_1(n) = \rho_1 u_d(n) \in \mathcal{U}_p$.

2) Let ρ_2 be the largest number such that $0 \leq \rho_2 \leq 1$ and $u_2(n) = u(n-1) + \rho_2(\rho_1 u_d(n) - u(n-1)) \in \mathcal{U}_r$. Let $u(n) = u_2(n)$.

The first step consists in scaling the control inputs to satisfy the position constraints. The second step consists in scaling the variation of the control inputs to satisfy the rate constraints. It was found useful to perform scaling for the position limits first, to give priority to the directionality of the commands.

Method 2: Relaxation of Control Requirements

The second method consists in relaxing the control requirements when the constraints are violated. In the context of the model reference control algorithm discussed here, the control requirements are primarily embedded in the constant k. For large values of k, the closed-loop poles are located far in the left-half plane, yielding tight and fast control. Conversely, in the presence of actuator saturation, a way to relax the control requirements is to reduce the value of the constant k. Because the control law has the form

$$u_d = u_a + u_b, \qquad u_a = (\boldsymbol{CB})^{-1}k(r_M - y)$$
$$u_b = (\boldsymbol{CB})^{-1}(-\boldsymbol{CA}x - \boldsymbol{Cd}) \tag{11}$$

the method consists in finding the largest ρ such that $0 \leq \rho \leq 1$ and $\rho u_a + u_b \in \mathcal{U}$. With this observation in mind, the method is found to

require a scaling similar to that performed in algorithm 1. However, it is somewhat more difficult to code because of the additional term u_b. In algorithm 2, which follows, $u_{a,1}$ refers to the first component of the vector u_a.

1) Find the range $R_1 = [r_{1,\min}, r_{1,\max}]$ such that $u_{1,\min} \le \rho u_{a,1} + u_{b,1} \le u_{1,\max}$ for all $\rho \in [r_{1,\min}, r_{1,\max}]$. This range can be obtained by considering nine cases:

a) If $u_{b,1} \le u_{1,\min}$ and $u_{a,1} + u_{b,1} \ge u_{1,\max}$, let

$$\rho_{1,\min} = \frac{u_{1,\min} - u_{b,1}}{u_{a,1}}, \qquad \rho_{1,\max} = \frac{u_{1,\max} - u_{b,1}}{u_{a,1}}, \tag{12}$$

and $r_{1,\min} = \rho_{1,\min}$, and $r_{1,\max} = \rho_{1,\max}$.

b) If $u_{b,1} \le u_{1,\min}$ and $u_{1,\min} \le u_{a,1} + u_{b,1} \le u_{1,\max}$, let $r_{1,\max} = 1$ and $r_{1,\min} = \rho_{1,\min}$, where $\rho_{1,\min}$ is defined in Eq. (12).

c) If $u_{b,1} \le u_{1,\min}$ and $u_{a,1} + u_{b,1} \le u_{1,\min}$, then $R_1 = \emptyset$, the empty set.

d) If $u_{1,\min} \le u_{b,1} \le u_{1,\max}$ and $u_{a,1} + u_{b,1} \ge u_{1,\max}$, let $r_{1,\min} = 0$ and $r_{1,\max} = \rho_{1,\max}$, where $\rho_{1,\max}$ is defined in Eq. (12).

e) If $u_{1,\min} \le u_{b,1} \le u_{1,\max}$ and $u_{1,\min} \le u_{a,1} + u_{b,1} \le u_{1,\max}$, let $r_{1,\min} = 0$ and $r_{1,\max} = 1$.

f) If $u_{1,\min} \le u_{b,1} \le u_{1,\max}$ and $u_{a,1} + u_{b,1} \le u_{1,\min}$, let $r_{1,\min} = 0$ and $r_{1,\max} = \rho_{1,\min}$, where $\rho_{1,\min}$ is defined in Eq. (12).

g) If $u_{b,1} \ge u_{1,\max}$ and $u_{a,1} + u_{b,1} \ge u_{1,\max}$, then $R_1 = \emptyset$, the empty set.

h) If $u_{b,1} \ge u_{1,\max}$ and $u_{1,\min} \le u_{a,1} + u_{b,1} \le u_{1,\max}$, let $r_{1,\max} = 1$ and $r_{1,\min} = \rho_{1,\max}$, where $\rho_{1,\max}$ is defined in Eq. (12).

i) If $u_{b,1} \ge u_{1,\max}$ and $u_{a,1} + u_{b,1} \le u_{1,\min}$, let $r_{1,\min} = \rho_{1,\max}$ and $r_{1,\max} = \rho_{1,\min}$, where $\rho_{1,\min}$ and $\rho_{1,\max}$ are defined in Eq. (12).

2) Repeat step 1 with appropriate index changes to determine the ranges R_2 and R_3.

3) Find $R = R_1 \cap R_2 \cap R_3$. If R is not empty, let ρ be the largest number in R and $u = \rho u_a + u_b$. If R is empty, let u be such that, for $i = 1, \ldots, 3$, $u_i = u_{i,\min}$ if $u_{d,i} < u_{i,\min}$, $u_i = u_{i,\max}$ if $u_{d,i} > u_{i,\max}$, and $u_i = u_{d,i}$ otherwise.

The tests are based on the values of $u_{b,1}$ and $u_{a,1} + u_{b,1}$, which are the values of $\rho u_{a,1} + u_{b,1}$ for $\rho = 0$ and $\rho = 1$. The variables $\rho_{1,\min}$ and $\rho_{1,\max}$ are the values of ρ such that $\rho u_{a,1} + u_{b,1}$ are equal to $u_{1,\min}$ and $u_{1,\max}$, respectively. These values of ρ may be outside the range [0, 1], but are only calculated if inside the interval. The specific implementation is chosen to avoid a possible division by zero in Eq. (12).

One difficulty with the concept is that it is possible for no ρ to exist. In such a case, a simple saturation function is applied (in step 3). This case was found to occur occasionally in simulations, although not frequently.

The method can be applied to control laws other than the model reference control law. For a linear quadratic control law, a similar method can be implemented by raising the penalty on the control deviations when the control constraints are violated. For the model reference control law, however, determination of the maximum constant ρ is relatively easy.

Method 3: Scaling of Reference Inputs

The third approach is similar to the first procedure, in that an input signal is scaled. However, the reference input r_M is scaled instead of the control input u_d. It makes sense to scale the pilot inputs instead of the control inputs for the directionality of the pilot commands to be preserved. Computationally, the method is close to the second method because one may write the control input as

$$u_d = u_a + u_b$$
$$u_a = (CB)^{-1} k r_M, \qquad u_b = (CB)^{-1}(-CAx - Cd - ky) \tag{13}$$

and the same procedure can be applied as in algorithm 2 to find a ρ such that $\rho u_a + u_b \in \mathcal{U}$. One difficulty, however, is that it is far easier to encounter a case where no ρ exists. A particular situation where this occurs is when the reference input moves from a large value to a zero value, so that the scaling of the reference input is ineffective in avoiding saturation. To resolve the problem, the concept may be modified so that a linear combination of the previous reference input and the current reference input is used. Instead of replacing r_M by ρr_M, the procedure replaces $r_M(n)$ by $\rho r_M(n) + (1-\rho) r_M(n-1)$. For $\rho = 1$, the modified reference input is equal to the reference input. For $\rho = 0$, the modified reference input is equal to the previous reference input. Interestingly, the same portion of algorithm 2 can again be applied, but with different definitions of u_a and u_b. Specifically, note that $u_d = u_a + u_b$, with

$$u_a = (CB)^{-1} k r_M(n) - (CB)^{-1} k r_M(n-1)$$
$$u_b = (CB)^{-1}(-CAx - Cd - ky) + (CB)^{-1} k r_M(n-1) \tag{14}$$

Algorithm 3, which results, is described next.

1) Find the largest ρ such that $0 \le \rho \le 1$ and $\rho u_a + u_b \in \mathcal{U}$, using the method of algorithm 2, but with u_a and u_b defined in Eq. (13). If a ρ is found, let $u = \rho u_a + u_b$ and stop; otherwise proceed.

2) Find the largest ρ such that $0 \le \rho \le 1$ and $\rho u_a + u_b \in \mathcal{U}$, using the method of algorithm 2, but with u_a and u_b defined in Eq. (14). If a ρ is found, let $u = \rho u_a + u_b$ and replace $r_M(n)$ by $\rho r_M(n) + (1-\rho) r_M(n-1)$. If no ρ can be found, let u be such that, for $i = 1, \ldots, 3$, $u_i = u_{i,\min}$ if $u_{d,i} < u_{i,\min}$, $u_i = u_{i,\max}$ if $u_{d,i} > u_{i,\max}$, and $u_i = u_{d,i}$ otherwise.

If the states and the parameters vary by a small amount over a sampling interval, the method will produce a value of ρ such that the control signal is admissible. Otherwise, it is possible that no ρ could be found, and the procedure again reverts to a saturation function in such a case.

Method 4: Least-Squares Approximation of Commanded Accelerations

This method consists in approximating the accelerations that would be produced by the desired control inputs. Recall that y contains the pitch, roll, and yaw rates. Therefore, the equation

$$\dot{y} = CAx + CBu + CBd \tag{15}$$

shows that the acceleration produced by a control input u_d is CBu_d. The acceleration is the control moment divided by the inertia (in a matrix sense), so that the concept is not very different from the approximation of the control moments. The fourth method consists in finding an admissible control u such that the acceleration CBu optimally approximates the desired acceleration CBu_d (Ref. 4). The optimization criterion is a least-squares criterion, so that the objective is to find $u \in \mathcal{U}$, such that

$$e(u, u_d) = \|CBu - CBu_d\|^2 \tag{16}$$

is minimized. In the absence of constraints, the solution is obtained by setting

$$\frac{\partial}{\partial u}\|CBu - CBu_d\|^2 = 0 \tag{17}$$

which yields

$$\frac{\partial}{\partial u}(CBu - CBu_d)^T (CBu - CBu_d)$$
$$= 2(CB)^T (CBu - CBu_d) = 0 \tag{18}$$

In the case of an invertible CB matrix, this equation gives $u = u_d$. With constraints, the solution can be obtained in a similar way. The idea behind the algorithm is to consider all of the possible cases, which are such that either zero, one, two, or three constraints are active.

Algorithm 4 is next.

1) If $u_d \in \mathcal{U}$, let $u = u_d$ and stop; otherwise proceed.

2) For $u_1 = u_{1,\min}$, find u such that $e(u, u_d)$ is minimized without constraints on u_2 and u_3. To that effect, define $u_{2,3}$ to be the vector composed of the second and third elements of u, $(CB)_1$ to be the vector composed of the first column of CB, and $(CB)_{2,3}$ to be the 3×2 matrix composed of the second and third column of CB. Then, let the optimum u be given by $u_{1,\min}$, and

$$u_{2,3} = \left[(CB)_{2,3}^T (CB)_{2,3}\right]^{-1} (CB)_{2,3}^T \cdot [(CB)u_d - (CB)_1 u_{1,\min}] \tag{19}$$

Determine the value of $e(u, u_d)$.

3) Repeat step 2 with the appropriate change of indices to determine the optimal $u's$ constrained to $u_1 = u_{1,\max}$ and similarly for $u_2 = u_{2,\min}$, $u_2 = u_{2,\max}$, $u_3 = u_{3,\min}$, and $u_3 = u_{3,\max}$. Determine the value of $e(u, u_d)$ in every case.

4) For $u_1 = u_{1,\min}$ and $u_2 = u_{2,\min}$, find u such that $e(u, u_d)$ is minimized without constraints on u_3, that is,

$$u_3 = \left[(CB)_3^T (CB)_3\right]^{-1} (CB)_3^T \cdot [(CB)u_d - (CB)_1 u_{1,\min} - (CB)_2 u_{2,\min}] \tag{20}$$

Determine the value of $e(u, u_d)$.

5) Repeat step 4 with the appropriate change of indices to determine the optimal $u's$ constrained to $u_1 = u_{1,\min}$ and $u_2 = u_{2,\max}$ and similarly for every pair among $u_1 = u_{1,\min}, u_1 = u_{1,\max}, u_2 = u_{2,\min}$, $u_2 = u_{2,\max}$, $u_3 = u_{3,\min}$, and $u_3 = u_{3,\max}$. Determine the value of $e(u, u_d)$ in every case.

6) Determine the value of $e(u, u_d)$ for $u_1 = u_{1,\min}$, $u_2 = u_{2,\min}$, and $u_3 = u_{3,\min}$ and similarly for every triplet among $u_1 = u_{1,\min}$, $u_1 = u_{1,\max}$, $u_2 = u_{2,\min}$, $u_2 = u_{2,\max}$, $u_3 = u_{3,\min}$, and $u_3 = u_{3,\max}$.

7) Collect all of the candidate $u's$, eliminate those that do not belong to $\mathcal{U}$, and select the one that yields the smallest $e(u, u_d)$.

The computation of $u_{2,3}$ is based on

$$(CB)u - (CB)u_d = (CB)_{2,3}u_{2,3} + (CB)_1 u_{1,\min} - (CB)u_d \tag{21}$$

and using a similar derivation as in Eq. (18). Alternatively, the method of Lagrange multipliers can be used to find another form of the solution. Specifically, a constraint

$$e_1^T u = u_{1,\min} \tag{22}$$

is imposed, where e_1 is a vector that is zero, except for the first component, which is 1. Extending Eq. (18), the following equation is obtained:

$$u = [(CB)^T (CB)]^{-1}\left[(CB)^T (CB)u_d - (\lambda/2)e_1\right] \tag{23}$$

where λ is a Lagrange multiplier. Using Eq. (22), the following result is obtained:

$$u = u_d - \frac{e_1^T u_d - u_{1,\min}}{e_1^T (CB)^{-1}[(CB)^{-1}]^T e_1} \cdot (CB)^{-1}[(CB)^{-1}]^T e_1 \tag{24}$$

This equation is an alternative to Eq. (19). Normally, this equation would not be advantageous, because it requires the inverse of the 3×3 matrix CB, instead of the 2×2 matrix $(CB)_{2,3}^T (CB)_{2,3}$. However, assuming that $(CB)^{-1}$ is calculated for the determination of u_d, no new inverse is needed, and this alternative is useful. For u_3, a similar procedure can be used, but it is not advantageous because u_3 only requires the inverse of a scalar in Eq. (20), and the expression equivalent to Eq. (24) in that case is more complicated. Some steps can also be eliminated. Specifically, step 4 is not necessary if, in step 2, an admissible u was obtained. Similar cases can be bypassed in steps 5 and 6, depending on results obtained in preceding steps.

Comparison of the Methods and Alternatives

The first method is the easiest to implement. Its computation does not depend on the estimates of the plant matrices, so that the result is not affected by errors in the estimation procedure. The second and third methods are trickier to code, but the computations are still simple. The concepts that they implement are more appealing, but a disadvantage is that there may be cases where no solution exists. The problem can be resolved by using a least-squares approximation, such as in method 4. The algorithm, however, is more computationally demanding. Methods 2–4 exploit the knowledge of the plant parameters and, therefore, of the generated moments. This knowledge would be expected to improve performance if the estimates of the plant parameters are good, but not in the transient following a failure.

The methods highlight different possibilities for the choice of a command-limiting method. The relaxation of the control requirements is the closest to a control law design that formally accounts for actuator saturation. In general, this concept leads to a low-gain/high-gain control law, and instead of the model reference control law considered here, a linear quadratic control law with adjustable penalties may be used.[10] The implementation with the model reference control law, however, yields a particularly convenient solution.

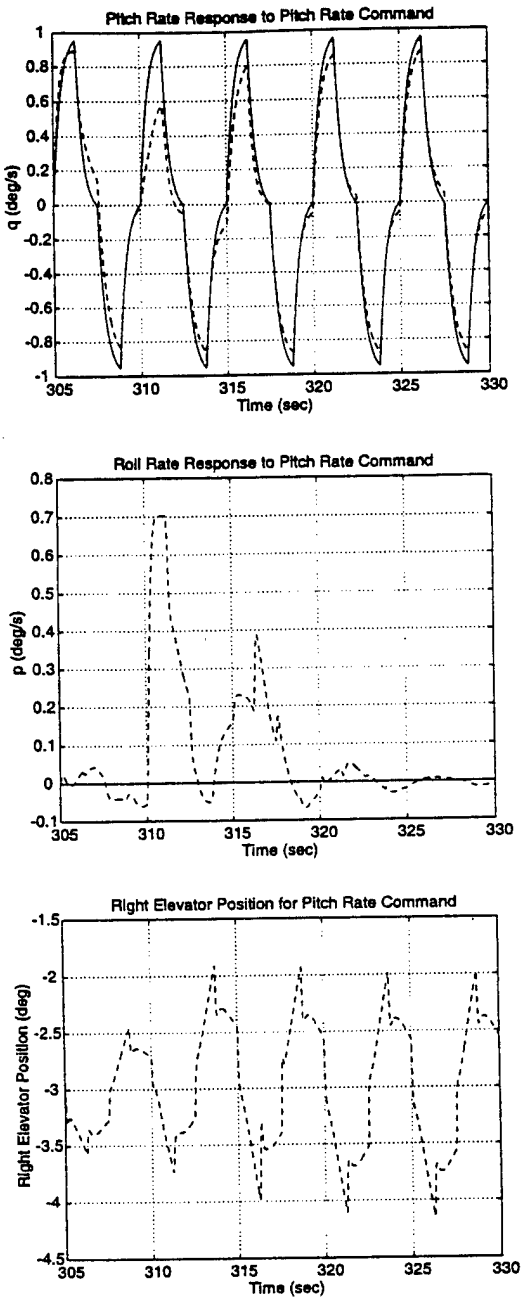
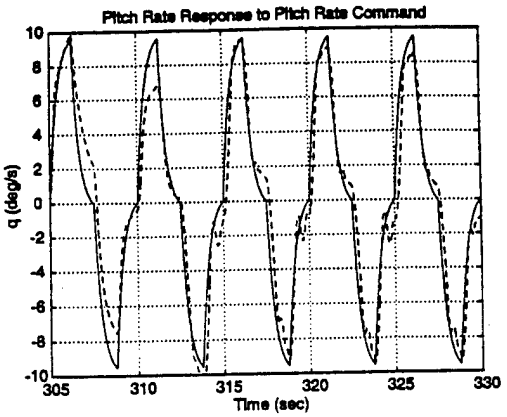
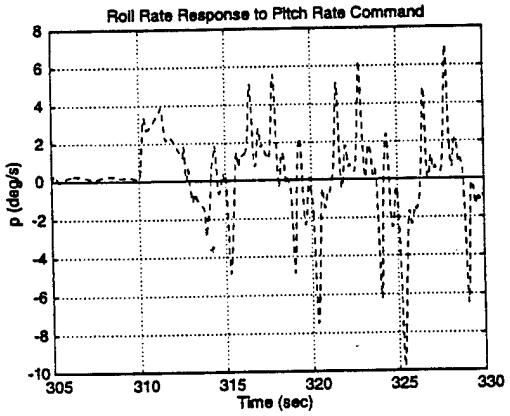


Fig. 1 Responses to a small pitch rate command: no command limiting.

Another concept represented in the four methods is the approximation, in some multi-dimensional space, of a desired input vector. Multiple choices of the vector are possible, including the control input, the reference input, and the commanded acceleration. The commanded moment is also a possibility, but it is not much different from the commanded acceleration. The reference input and the commanded acceleration are elegant choices, but the transformation between those signals and the control signals complicate the algorithms.

Given the choice of an input vector, approximation may be performed either through scaling or through minimization of some criterion. Scaling, e.g., methods 1–3, is easier to perform and places emphasis on directionality. However, a solution may not always exist. In that case, one must either return to a straight saturation or to the minimization of some criterion. The approach based on some optimization is possible, but is susbtantially more complex.





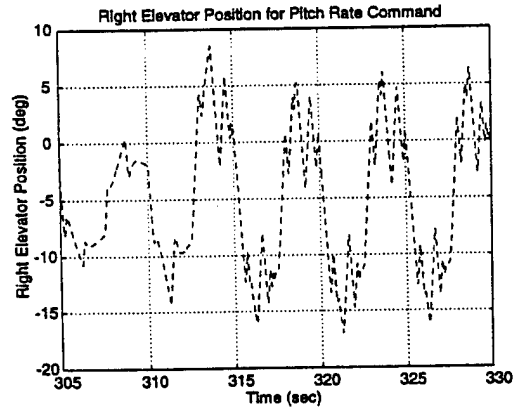
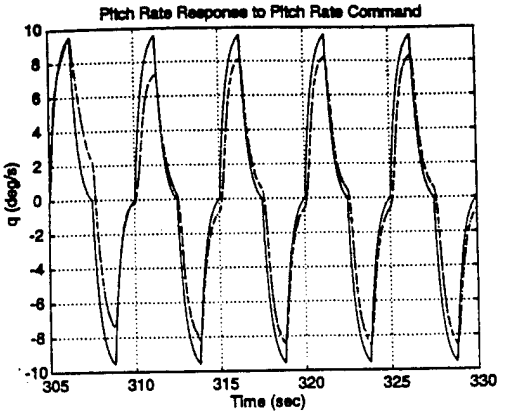
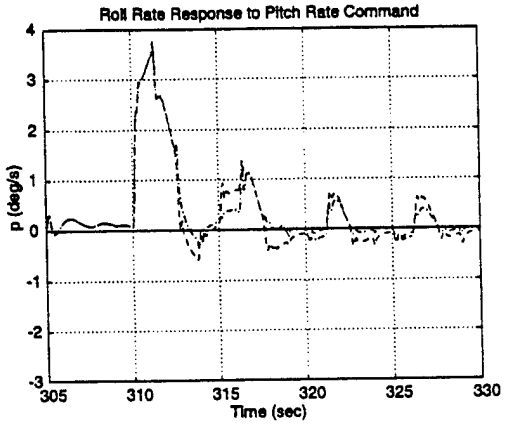


Fig. 2 Responses to a large pitch rate command: no command limiting.





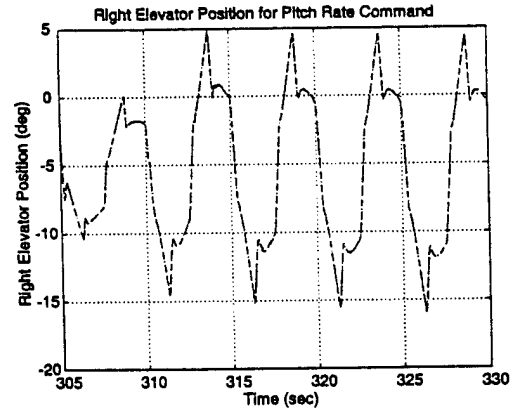


Fig. 3 Responses to a large pitch rate command: methods 1 and 2.

In that case, alternative criteria to the l_2 distance (least squares) are possible, such as l_1 and l_∞ distances.[5]

Prioritization may be useful, although is not considered in this paper. It is relatively straightforward to extend the least-squares algorithm to weight accelerations in various axes differently. Such a modification may be useful, for example, in the case where an axis is unstable. For the approaches based on scaling, it is also possible to prioritize axes by performing scaling with different coefficients in the different axes. An implementation of this concept can be found in Ref. 11 (in that case, an interesting idea that is used is to give different priorities to components of the control input associated with stability augmentation and maneuvering, instead of different axes).

Simulation Results

Simulation Parameters

Simulations were carried out using a detailed model of a twin-engine aircraft developed at NASA Dryden Flight Research Center.[12] The model is a detailed representation of the aircraft's nonlinear dynamics, including full envelope aerodynamics, atmospheric model, detailed engine dynamics, actuator dynamics, and saturation. In the code, the position limits are from 15 to -25 deg for the elevators, ± 20 deg for the ailerons, and ± 30 deg for the rudder. The rate limits are 24 deg/s for the elevators and for the ailerons. The rudder rate saturation was disabled in the original code and was left so. In the command limiting methods, the rudder limits were neglected accordingly. The dynamics of the actuators besides the saturation are those of first-order systems with poles at -20 rad/s. In the original code, there was also a cross feed between aileron command and antisymmetric elevator command, which was eliminated.

For the control law, the constant k in the reference model was set to 2.5 rad/s. In the adaptive algorithm, the constants were set to $\lambda = 0.99$, $\alpha = 10$, and $c = 0.1$. For the identification algorithm, the actuator signals that were used were the commanded signals, filtered by first-order systems. An alternative would have been to use the actual positions, as measured by synchros on a real airplane, but this option was not pursued.

Control Performance Without Command Limiting

The first set of simulation results is shown in Figs. 1 and 2 and was obtained without command limiting. The reference model output

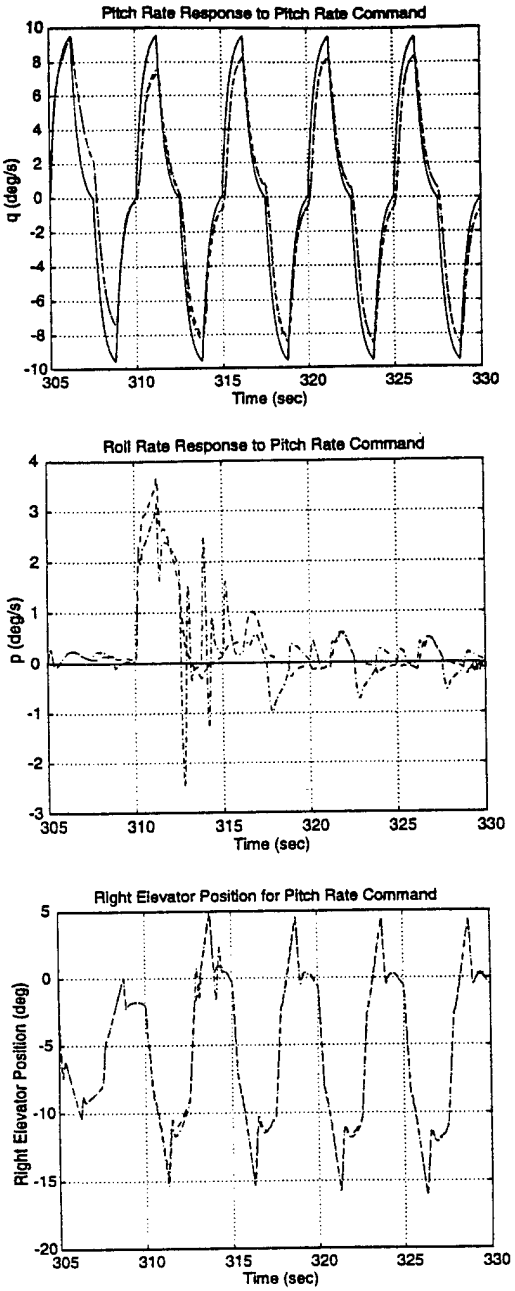


Fig. 4 Responses to a large pitch rate command: methods 3 and 4.

y_M is shown by a solid line. The output y is shown as a dashed line. The responses were obtained for a failed aircraft, with the failure being a locked left horizontal tail surface and occurring at $t = 310s$. The initial parameter values that were used in the recursive algorithm were the parameter matrices determined by a batch procedure for the unfailed aircraft.

Responses shown correspond to single-axis commands consisting in multiple step changes. Single-axis commands were chosen to assess the capability of the algorithm to maintain decoupling despite the loss of symmetry of the aircraft after failures. Multisteps were applied to observe the tuning of the responses by the adaptive algorithm.

The plots in Fig. 1 show the responses to a small pitch rate command of 1 deg/s. From top to bottom, the plots show the pitch rate response, the roll rate response, and the response of the right (unfailed) elevator. The results show that the adaptive algorithm is able to reconfigure the control law so that trim, tracking, and decoupling of the axes are all successfully achieved after a brief transient period. The elevator response shows that the magnitude of the elevator deflection required to produce the pitching moment is doubled after the failure, as expected.

In Fig. 2, the responses are shown for a pitch rate command of 10 deg/s, instead of 1 deg/s. In that case, the pitch rate response exhibits some ringing, and the roll rate response shows a large coupling. This coupling does not decrease with time. The actuator positions do not reach the limits, but the rate limits are reached most of the time. The rate limits manifest themselves as the linear portions of the elevator responses and are the source of a considerably degraded performance in both axes. The problem is that the actuators controlling different axes do not saturate in a coherent way, so that decoupling is lost. As we will see, the issue is not one of insufficient control power, but of proper usage of the control power available.

Control Performance with Command Limiting

The responses with the command limiting are shown in Figs. 3 and 4. In Fig. 3 are the responses for methods 1 and 2, shown as dashed lines and dot–dashed lines, respectively. In Fig. 4 are the responses for methods 3 and 4, also shown as dashed lines and dot–dashed lines (respectively). The pitch rate responses are much improved with respect to the original control law, with the ringing being eliminated. In the roll rate responses, the cross coupling is much reduced and decreases with time. Performance is comparable for all methods, although some transient oscillations appear with method 4. The similar performance of the four methods is surprising inasmuch as their principles and their relative complexities are so different. A possible explanation is that all four methods manage to maintain the directionality of the commands, something simple saturation of the controls fails to do.

The adaptive algorithm with the methods of command limiting is also able to handle multiple failures. In Fig. 5, both the left aileron and the left stabilator are locked and method 4 is used. The responses shown are for a roll rate command of 30 deg/s and are found to be good, despite the significant loss of control power.

In some instances, it was found that the transient behavior of method 4 was degraded. For example, Fig. 6 shows the responses for a roll rate command of 50 deg/s and a locked aileron failure. The bottom plot shows the aileron command (instead of the elevator command shown in earlier plots). As in Fig. 4, the responses

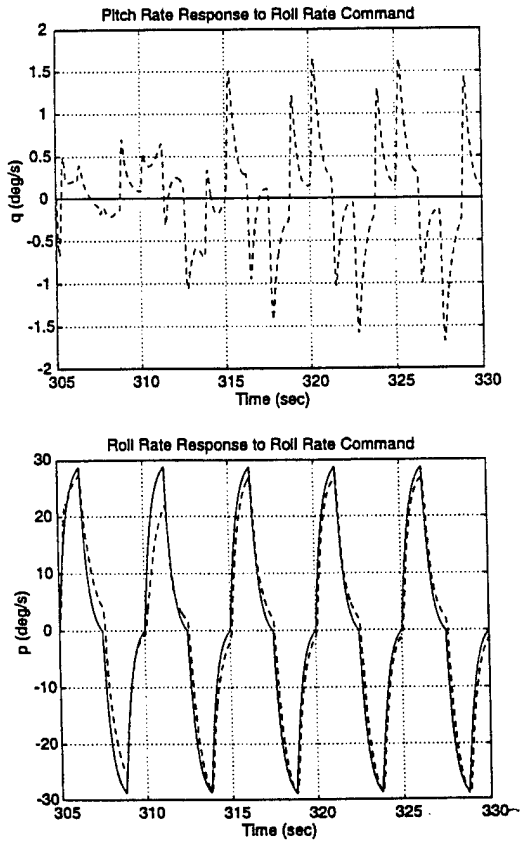


Fig. 5 Responses to a large roll rate command with locked stabilator and aileron: method 4.

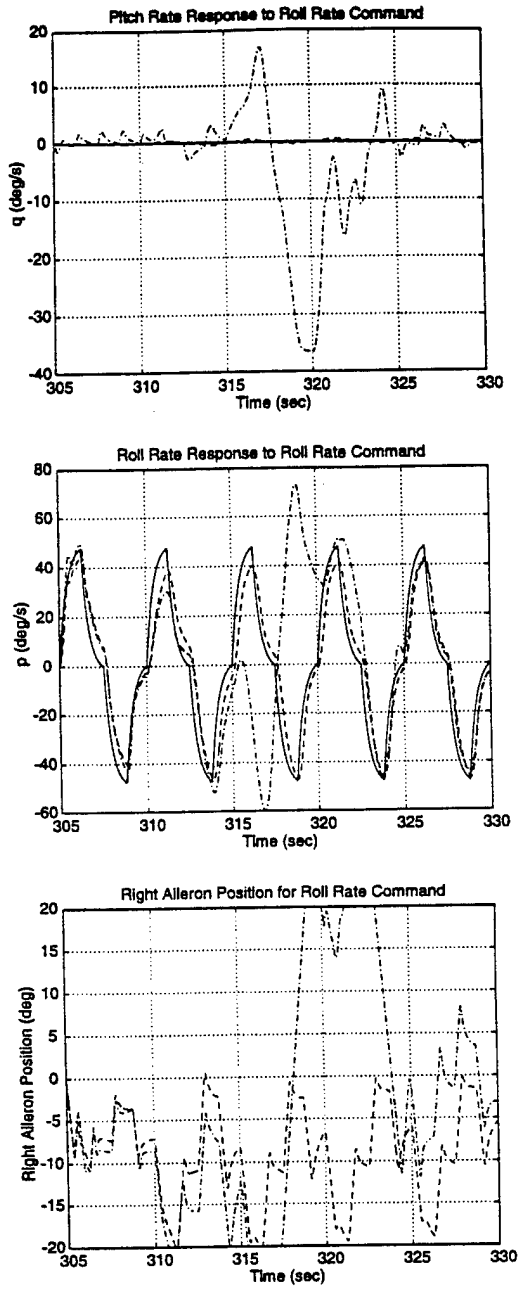


Fig. 6 Responses to a large roll rate command with locked aileron: methods 3 and 4.

for methods 3 and 4 are shown as dashed lines and dot–dashed lines, respectively. Large pitch and roll rate transients are observed for method 4. These transients are not present in the other three methods and are not observed for any of the four methods for a smaller reference input of 30 deg/s. The large transients (as well as the ringing observed in Fig. 4) are due to the uncertainty in the aircraft parameters after the failure because steady-state performance is comparable for all methods.

It was found that the command-limiting methods were used about 20% of the time in the runs shown in Figs. 3 and 4. For method 2, the no-ρ case was encountered 2% of the time, i.e., 10% of the time the command-limiting routine was called. For method 3, the no-ρ case was encountered about 0.4% of the time.

Effect of Actuator Dynamics

As pointed out by Bolling,[13] actuator dynamics may significantly reduce the rate of variation achieved under digital control. The effect is particularly important when the sampling rate is close to the time constants of the actuators. In the simulations reported in this paper, such effects were not observed, although the actuator model[12] includes first-order dynamics in addition to position and rate limits. An important consideration is that the control law discussed here does not rely on actuator positions to adjust the control increments (as was assumed in Bolling's thesis[13]), but rather uses previous actuator commands. Because the control law is not aware of the reduced response due to actuator dynamics, actuators are effectively overdriven. This result is achieved similarly to a suggestion made by Bolling for this problem, but without any adjustment required in the control law. Another reason why the simulations shown in this paper do not exhibit the effects of actuator dynamics discussed by Bolling is that the model[12] relies on a simple Euler integration rule, with the integration step equal to the sampling rate of the control law. Therefore, fine characteristics of the intersample behavior are not represented in the simulations.

Conclusions

Four methods of command limiting were studied, spanning from a simple scaling of the control inputs to a least-squares approximation of commanded accelerations. The emphasis was on problems occurring in the context of reconfigurable flight control. Simulations showed that, without command limiting, considerable degradation of performance could result from large reference inputs. These problems could be alleviated using the command-limiting methods. Overall, the performance was found to be comparable for all methods. This was somewhat surprising given the different concepts that were implemented and the different complexity levels.

Sometimes, degraded transient responses were observed with the least-squares approximation of commanded acceleration. This degradation may be because the method uses the knowledge of the generated moments and because there is considerable uncertainty in those moments for a period following a failure.

An important problem, which was not addressed in this paper, is that of allocation of control authority among a number of redundant actuators. The least-squares approximation of the commanded acceleration can be extended to that problem, without conceptual difference. Computational requirements, however, grow rapidly with the number of actuators. Another approach is based on the scaling of the variation of commanded acceleration, which may be viewed as an extension of the first method, although it is conceptually more sophisticated and computationally more complex.

Stability issues were not addressed and are a significant area of interest, especially for unstable aircrafts. Because the methods of command limiting presented do not affect the control signals away from the limits, the stability properties for small motions are the same as those of the linear control law. For large motions that induce control saturation, stability guarantees are extremely difficult to obtain. Because stability is also affected by pilot dynamics, such issues are likely to be resolved in practice through extensive flight simulations and testing.

Acknowledgments

This effort was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, U.S. Air Force, Grants F49620-95-1-0341 and F49620-97-1-0405. The views and conclusions contained in the paper are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government.

References

[1]Durham, W. C., "Constrained Control Allocation," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 4, 1993, pp. 717–725.

[2]Bordignon, K. A., and Durham, W. C., "Closed-Form Solutions to the Constrained Control Allocation Problem," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 5, 1995, pp. 1000–1007.

[3]Durham, W. C., and Bordignon, K. A., "Multiple Control Effector Rate Limiting," *Journal of Guidance, Control, and Dynamics*, Vol. 19, No. 1, 1996, pp. 30–37.

[4]Honeywell Technology Center and Lockheed Martin Skunk Works, "Multivariable Control Design Guidelines," Rept. WL-TR-96-3099, Wright–Patterson AFB, OH, May 1996.

[5]Pachter, M., Chandler, P. R., and Mears, M., "Reconfigurable Tracking Control with Saturation," *Journal of Guidance, Control, and Dynamics*, Vol.

18, No. 5, 1995, pp. 1016–1022.

[6]Bodson, M., and Groszkiewicz, J., "Multivariable Adaptive Algorithms for Reconfigurable Flight Control," *IEEE Transactions on Control Systems Technology*, Vol. 5, No. 2, 1997, pp. 217–229.

[7]Ward, D. G., and Barron, R. L., "A Self-Designing Receding Horizon Optimal Flight Controller," *Proceedings of the American Control Conference* (Seattle, WA), Inst. of Electrical and Electronics Engineers, Piscataway, NJ, 1995, pp. 3490–3494.

[8]Bodson, M., "An Adaptive Algorithm with Information-Dependent Data Forgetting," *Proceedings of the American Control Conference* (Seattle, WA), Inst. of Electrical and Electronics Engineers, Piscataway, NJ, 1995, pp. 3485–3489.

[9]Bodson, M., and Pohlchuck, W., "Integral Compensation in Adaptive Algorithms for Reconfigurable Flight Control," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (San Diego, CA), AIAA, Reston, VA, 1996 (AIAA Paper 96-3772).

[10]Lin, Z., Pachter, M., Banda, S., and Shamash, Y., "Feedback Design for Robust Tracking of Linear Systems with Rate Limited Actuators," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (New Orleans, LA), AIAA, Reston, VA, 1997, pp. 780–788.

[11]Buffington, J. M., "Tailless Aircraft Control Allocation," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (New Orleans, LA), AIAA, Reston, VA, 1997, pp. 737–747.

[12]Brumbaugh, R. W., "Aircraft Model for the AIAA Controls Design Challenge," *Journal of Guidance, Control, and Dynamics*, Vol. 17, No. 4, 1994, pp. 747–752.

[13]Bolling, J. G., "Implementation of Constrained Control Allocation Techniques Using an Aerodynamic Model of an F-15 Aircraft," M.S. Thesis, Aerospace and Ocean Engineering Dept., Virginia Polytechnic Inst. and State Univ., Blacksburg, VA, May 1997.

FAST IMPLEMENTATION OF DIRECT ALLOCATION AND EXTENSION TO COPLANAR CONTROLS

John A. M. Petersen and Marc Bodson*

Department of Electrical Engineering

University of Utah, Salt Lake City, UT 84112

(801) 581 8590 bodson@ee.utah.edu

Abstract

The paper considers the direct allocation method proposed by Durham. The original method assumed that any three columns of the controls effectiveness matrix were linearly independent. In this paper, the condition is relaxed, so that systems with coplanar controls can be considered. For fast on-line execution, an approach using spherical coordinates is also presented and results of the implementation are reported. Linearized state-space models of a C-17 aircraft and of a tailless aircraft are used in the evaluation.

---

* Marc Bodson is a Professor of Electrical Engineering and a Senior Member, AIAA. John Petersen is a graduate student. Effort sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grants F49620-97-1-0405 and F49620-98-1-0013. The views and conclusions contained in the paper are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government.

1. Introduction

In order to increase the reliability of aircrafts, configurations with a large number of actuators and control surfaces are advantageous. Reconfigurable control laws may be used to exploit all the available control power despite failures and damages.[1,2] *Control allocation* is the problem of distributing the control requirements among multiple actuators in order to satisfy the desired objectives while accounting for the limited range of the actuators. Although solutions exist for the control allocation problem, an issue of current interest is that of the feasibility of their implementation on existing computers for aircrafts with a large number of actuators.[3]

The *direct allocation* approach[4,5,6] is based on the concept of the *attainable moment set* (AMS), which is the set of all the moment vectors that are achievable within the control constraints. The method of direct allocation allows one to achieve 100% of the AMS, whereas other approaches such as daisy chaining, pseudo-inverse and generalized inverse solutions have been shown to achieve a smaller volume.[7]

In the direct allocation method, the moment vectors are assumed to be related to the controls through the linear transformation $m = CBu$, where m is the resultant moment, u is the set of controls, and CB is referred to as the controls effectiveness matrix. The original method for three-moments developed by Durham was restricted to systems in which any three columns of CB are linearly independent. For this case, the boundary of the AMS consists of parallelograms defined by pairs of controls varying between their limits. As it turns out, the control needed to produce any moment on the boundary of the AMS is unique. In the direct allocation method, moments lying inside the boundary of the AMS are obtained by scaling the controls required to produce a moment of maximum magnitude in the same direction. In a similar manner, moments lying outside the boundary are scaled down to the achievable values. Therefore, controls are always uniquely defined.

If the restriction on CB is not satisfied, then the boundary of the AMS is defined by polygons rather than parallelograms, and each facet is bounded by $2p$ sides, where p is the number of controls defining the polygonal facet. With more than two variables describing the facet, the solution is not always unique, even on the boundary of the AMS. Because this situation occurs when the effects of three or more controls are linearly dependent in a three-dimensional space, the terminology '*coplanar controls*' is introduced to explicitly refer to this case. Systems with

coplanar controls are loosely called '*coplanar systems.*' The geometry of the AMS boundary is further described in the paper, and a possible choice for the selection of the control is proposed, given the non-uniqueness properties.

Next, we consider that most of the computational burden in using the direct allocation method lies in finding the facet in which the desired moment resides. Generally, computations may be split into off-line and on-line computations. Off-line computations are defined to be those that may be performed at the design stage or, in the case of a reconfigurable control law, at a slower rate than the normal sampling rate. On-line computations are those that are required for the determination of the control input at every sampling instant. A significant portion of the computations may be performed off-line in the direct allocation method, and consist in the determination of the set of attainable moments. On-line computations include the search for the facet in the attainable moment set that is aligned with the desired moment, and the determination of the control input using appropriate scaling.

To reduce the on-line computations, a representation of the AMS in 2-dimensional space, using spherical coordinates, is shown to be beneficial. The new method converts the AMS representation into a two-dimensional system, where special techniques can be used to accelerate the search. Two options are suggested for the implementation. The first method computes facet boundaries that are used on-line to rapidly eliminate a large number of facets from the search. The second method creates a two-dimensional array relating the spherical coordinates of the desired moment to a corresponding facet identifier. The appropriate facet is found on-line by table look-up, requiring no iterations and virtually no computations. The spherical methods are also developed for coplanar systems. Rather than using polygonal facets for the rapid search, a representation using multiple coplanar sub-facets is considered. Examples used to illustrate the concepts proposed include a C-17 aircraft model with 16 actuators and an advanced tailless fighter model with 11 actuators.

2. Problem Statement

Consider the linearized aircraft model

$$\dot{x} = Ax + Bu$$
$$y = Cx \tag{1}$$

where $x \in R^5$, $u \in R^n$, $y \in R^3$. The states of the aircraft are given by x, and include the angle of attack, the pitch rate, the angle of sideslip, the roll rate, and the yaw rate. The output y contains the pitch rate, the roll rate, and the yaw rate. The control input, u, is constrained to limits

$$u_{i,\min} \leq u_i \leq u_{i,\max} \qquad \text{for } i = 1 \ldots n$$

The matrix B specifies the forces and moments generated by the actuators. These forces and moments are limited by the allowable range of control inputs. Since we are interested in controlling the output y, we consider the derivative of y, which is given by

$$\dot{y} = CAx + CBu \tag{2}$$

Model reference control laws[8] and dynamic inversion control laws[9] allow one to specify the trajectories of the output of the system by selecting the value of the term CBu due to the control input. The control allocation problem is then stated as follows:

Objective: Given a desired vector m_d, find the vector u such that CBu is closest to m_d in magnitude, with u satisfying the constraints and CBu proportional to m_d.

In the original formulation of Durham, the vector m_d was a desired moment. Here, the vector represents three desired rotational accelerations. We will nevertheless continue to refer to the set of achievable *CBu's* as the AMS.

3. Set of Attainable Moments and Direct Allocation

Initially, we make the following assumption:

Assumption (non-coplanar controls): Every 3x3 sub-matrix of CB is full rank.

Under this assumption, the following properties are obtained.

3.1. Properties of the AMS

The AMS is a convex polyhedron, whose boundary is the image of the facets of the control space. A facet of the control space is defined as the set obtained by taking all but two controls at their limits, and varying the two 'free' controls within the limits. A 2D facet in control space is rectangular. The projection of such a facet to moment space is a linear transformation resulting in a 2D parallelogram in 3D space. When any three columns of the *CB* matrix are linearly independent, every facet on the boundary of the AMS originates from a unique facet on the boundary of the control space. There are $2^{n-2}n!/[2!(n-2)!]$ facets in the control space. However, most of these facets map to the interior of the AMS, and the boundary of the AMS is comprised of only $n(n-1)$ facets[6]. The four corners of each facet of the AMS are called *vertices*, and the four sides are called *edges*. There are $n(n-1)+2$ vertices in the AMS.

3.2. Computation of the AMS

The boundary of the AMS is made of facets corresponding to all the possible pairs of input variables. For each pair, there is a multitude of facets in the original control space, but only two of them map to the boundary of the AMS. They may be found by looking for the combination of the other controls that maximizes the distance between the two facets. Hereafter, we refer to one of the facets as a 'max' facet and the other as a 'min' facet. The collection of all these pairs of facets then constitutes the boundary of the AMS.

To further explain the procedure, let *CB* be subdivided as

$$CB = [cb_1\ cb_2\ \ldots\ cb_n],$$

where cb_i is a column vector, and $m_i = cb_i\, u_i$ is the moment vector corresponding to the single control u_i. For a pair of controls, (u_i, u_j), $i \in \{1\ldots n\}$, $j \in \{i+1\ldots n\}$, let the normal to the plane of the facet, η_{ij}, be defined by taking the cross-product of the two vectors defining the facet

$$\eta_{ij} = cb_i \times cb_j \tag{3}$$

Then, the two farthest facets are determined through the two vectors

$$m_{\max} = \sum_{k=1,k\neq i,j}^{n} \mu_{k,\max} \tag{4}$$

$$\text{where} \quad \mu_{k,\max} = \begin{cases} cb_k u_{k,\max} & if\ (cb_k)^T \eta_{ij} > 0 \\ cb_k u_{k,\min} & if\ (cb_k)^T \eta_{ij} < 0 \end{cases}$$

and

$$m_{\min} = \sum_{k=1,k\neq i,j}^{n} \mu_{k,\min} \tag{5}$$

$$\text{where} \quad \mu_{k,\min} = \begin{cases} cb_k u_{k,\max} & if\ (cb_k)^T \eta_{ij} < 0 \\ cb_k u_{k,\min} & if\ (cb_k)^T \eta_{ij} > 0 \end{cases}$$

Note that the case where the vector product $(cb_k)^T \eta_{ij}$ is identically zero is impossible by virtue of the assumption of linear independence of any three columns of the *CB* matrix. However, this case is possible with coplanar systems and is discussed in detail in section 3.7. In coding this procedure, it is convenient to store an array of flags indicating the control values (max, min, or *free*) associated to each facet. A facet may also be assigned a number to index the array.

The vertices of the two facets are determined by using the maximum and minimum values of the other two *free* controls. For instance, the vertices for the max facet are

$$\begin{aligned} m_1 &= m_{\max} + cb_i u_{i,\min} + cb_j u_{j,\min} \\ m_2 &= m_{\max} + cb_i u_{i,\min} + cb_j u_{j,\max} \\ m_3 &= m_{\max} + cb_i u_{i,\max} + cb_j u_{j,\min} \\ m_4 &= m_{\max} + cb_i u_{i,\max} + cb_j u_{j,\max} \end{aligned} \tag{6}$$

Figure 1 shows the results of this procedure for a C-17 aircraft model. A 3-dimensional view of the boundary is shown. The facets are shaded according to height in the yaw acceleration

axis. The C-17 model includes sixteen separately controlled surfaces: 4 elevators, 2 ailerons, 2 rudders, and 8 spoilers. The set is delimited by 240 facets.

3.3. Computation of the Control Input

The control input is obtained by scaling the desired moment so that the scaled vector reaches the boundary of the AMS. On the boundary, there is a unique relationship between the moment and the value of the input needed to achieve it. If the desired moment is larger than the one attainable in the given direction, the moment vector is scaled to the achievable value. If the desired moment is smaller, the control input associated with the maximum attainable moment is scaled to obtain the desired moment.

The algorithm proceeds as follows. For a given facet, a basis spanning the moment space is formed by using the vector from the origin to one vertex of the facet and the vectors from this vertex to the two adjacent vertices. Let m_{base} be the vector to one of the vertices and m_i and m_j be the vectors to the other two vertices.

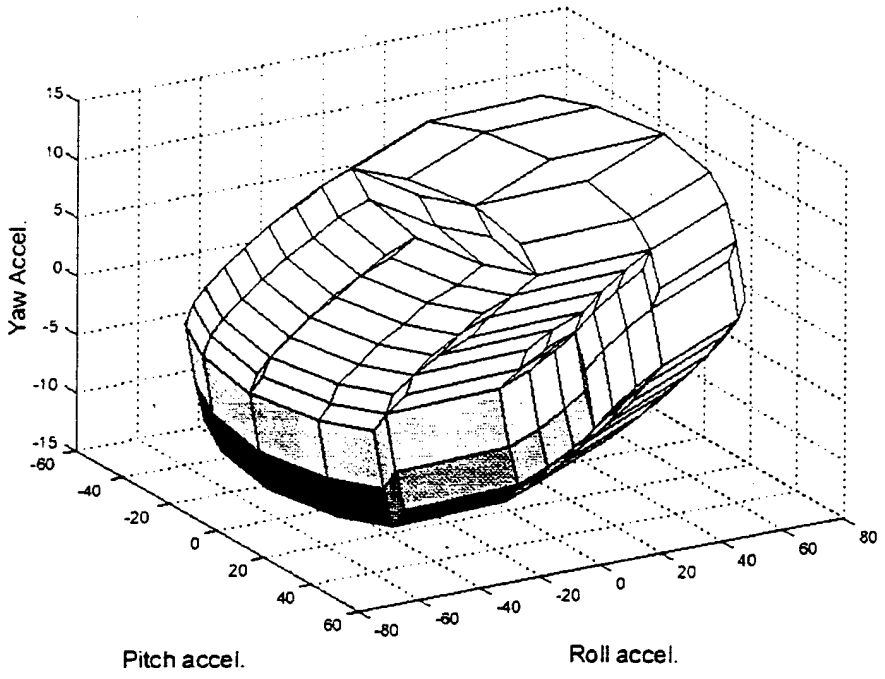


Figure 1. Set of attainable moments for a C-17 model.

Using m_d as the desired moment, we have

$$\begin{aligned} m_{base} &= m_1 \\ m_i &= m_4 - m_1 = cb_i \left(u_{i,\max} - u_{i,\min} \right) \\ m_j &= m_2 - m_1 = cb_j \left(u_{j,\max} - u_{j,\min} \right) \\ \rho_3 m_d &= \rho_1 m_i + \rho_2 m_j + m_{base} \end{aligned} \tag{7}$$

The free parameters ρ_1, ρ_2, ρ_3 are found by solving the 3x3 set of linear equations

$$\begin{bmatrix} \rho_1 \\ \rho_2 \\ \rho_3 \end{bmatrix} = \begin{bmatrix} -m_i & -m_j & m_d \end{bmatrix}^{-1} m_{base} \tag{8}$$

Figure 2 shows graphically the moment vector equation. The value $\rho_3 m_d$ is the point at which the vector m_d intersects the facet. The values of (ρ_1, ρ_2, ρ_3) determine whether m_d intersects the facet. If $\rho_3 > 0$, and ρ_1 and ρ_2 are both between 0 and 1, then a vector in the direction of the desired moment intersects the facet defined by m_{base}, m_i, and m_j.

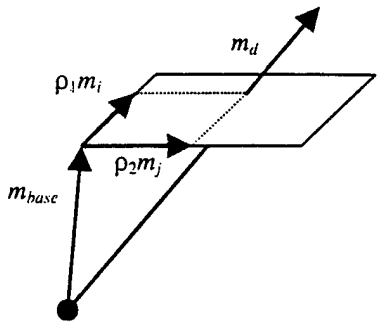


Figure 2. A desired moment intersecting a facet, with basis vectors shown in relation to the facet.

The control vector at the boundary is

$$u_{boundary} = u_{base} + \rho_1 u_i + \rho_2 u_j \tag{9}$$

where u_{base}, u_i, and u_j are the sets of controls which determine m_{base}, m_i, and m_j respectively. $u_{boundary}$ is the control associated to the maximum moment in the direction of the desired moment and within the control constraints. If $\rho_3 < 1$, the desired moment exceeds the maximum available moment and $u_{boundary}$ is taken to be the control. If $\rho_3 > 1$, the control is scaled to match the moment requirement, with $u = u_{boundary}/\rho_3$.

3.4. Sequential Search for Direct Allocation

The computation of the control input involves the solution of a linear system of three equations in three unknowns, and the linear combination of three input vectors. If the correct facet is used, the computations are minor, and the resulting control input satisfies the limits. If the incorrect facet is used, the values of (ρ_1, ρ_2, ρ_3) exceed their limits, and the control input will not satisfy the constraints. The computation may be used as a test of whether the facet is the correct one. If all the facets are tested sequentially in this manner, the procedure may be used for control allocation. We will refer to this approach as the *sequential search* procedure.

The computations for this procedure may be separated into off-line and on-line computations. The off-line code creates a table containing the four vertices associated to each facet. The on-line code consists of retrieving the vertex data, computing the control, and checking its feasibility. Once the correct facet is encountered, computations stop. The search will be time-consuming if the number of facets is large. The sequential search was nevertheless implemented to provide a baseline for the evaluation of the benefits of the methods proposed. More intelligent search techniques have been proposed[5,10], but these were not implemented for this paper. Instead, the use of spherical coordinates is investigated to accelerate the search.

3.5. Properties of the AMS for Systems with Coplanar Controls

For systems with coplanar controls, a p-dimensional volume ($p \geq 2$) in control space maps into a 2D facet in moment space and has $2p$ sides. The facet becomes a polygon defined by p controls. Figure 3 shows the AMS for an advanced tailless fighter model[11]. According to reference (11), the output vector y is composed of modified rotational rates. Specifically, the components of y are the pitch rate, the stability axis roll rate, and a blend of sideslip and stability axis yaw rate. The advanced tailless fighter model includes eleven separately controlled surfaces consisting of

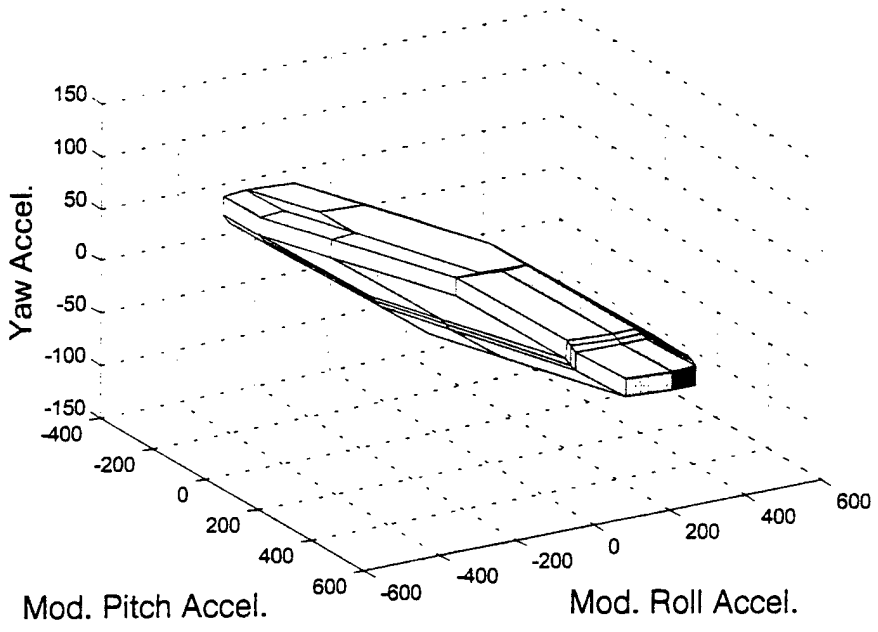


Figure 3: Set of attainable moments for an advanced tailless fighter model.

elevons, pitch flaps, thrust vectoring, outboard leading edge flaps, spoiler slot deflectors and all-moving tips. In this model, thrust pitch vectoring and the pitch flaps produce linearly dependent moments yielding coplanar controls with any third control variable. It was found that up to four control variables were coplanar $(2 \le p \le 4)$. Some polygonal facets are indeed clearly visible on the figure. The boundary of the AMS is delimited by 78 such facets.

3.6. Alternative Description for Coplanar Controls using Sub-Facets

Polygonal facets can also be described by a set of sub-facets that are the projections of the 2D facets of the control space. Each 2D facet is determined by two controls as in the non-coplanar case. For every pair of the p controls, there are 2^{p-2} identical sub-facets offset from each other in the same plane. Since there are $p(p-1)/2$ pairs of controls, the total number of sub-facets covering a polygonal facet is given by

$$n_{pf} = p(p-1)2^{p-3} \tag{10}$$

Figure 4 is a series of figures that details the relationship between a polygonal facet and its sub-

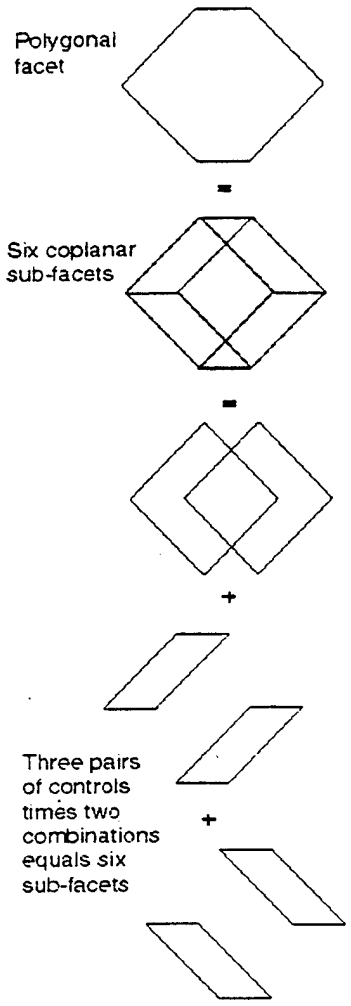


Figure 4. View of the relationship between a parent polygonal facet and its sub-facets.

facets. The top figure shows a polygonal facet. The middle figure shows how the six sub-facets cover the polygonal facet. The bottom figure shows the three sets of sub-facets in an exploded view. The representation of the AMS by sub-facets has been found to be more practical for the computation of the control input in the extension of the direct allocation method proposed here.

3.7. Computation of the AMS

The algorithm is similar to the one described in section 3.2. For every pair of controls, (u_i, u_j), $i \in \{1 \ldots n\}$, $j \in \{i+1 \ldots n\}$, each column of CB, excluding columns i and j, is evaluated. Coplanar controls can be determined by the product $(cb_k)^T \eta_{ij}$ where $k \neq i,j$. If this product is zero, then the k^{th} control is coplanar with the facet created by the control pair (u_i, u_j). The non-

coplanar controls (that correspond to the columns of *CB* that are linearly independent of columns *i* and *j*) are then selected so as to maximize the distance of the facet from the origin. As before, for each facet, there is an identical facet that lies on the opposite side of the AMS boundary and can be found using the opposite values for each of the *n-p* non-coplanar controls. Defining the set of indices

$$K = \left\{ \begin{matrix} i,\ j,\ \text{and indices of all controls} \\ \text{coplanar with } u_i \text{ and } u_j \end{matrix} \right\},$$

the maximum displacement vector is

$$m_{\max} = \sum_{\substack{k=1 \\ k \notin K}}^{n} \mu_{k,\max} \tag{11}$$

where $\mu_{k,\max}$ is defined in (4). The opposite facet is determined by the minimum displacement vector

$$m_{\min} = \sum_{\substack{k=1 \\ k \notin K}}^{n} \mu_{k,\min} \tag{12}$$

where $\mu_{k,\min}$ is defined in (5).

While the non-coplanar *n-p* controls determine the distance of a polygonal facet from the origin, the remaining *p* controls determine the shape of the polygonal facet. Each polygonal facet is made up of $t = \frac{p(p-1)}{2}$ sets of $r = 2^{p-2}$ sub-facets. Each sub-facet lies in the same plane, but has a different offset that shifts it with respect to the other sub-facets. The offset is determined by the sum of the *r* combinations of max and min control values of the coplanar controls. For every unordered pair of controls $\{(u_a, u_b) \mid a \in K, b \in K, a \neq b\}$, the offset of each sub-facet is computed using the controls $\{u_c \mid c \in K, c \notin \{a,b\}\}$ in different combinations of their upper and lower limits, resulting in

$$offset_q = cb_c u_c \quad \text{where } q \in \{1 \ldots r\}. \tag{13}$$

Since c is a vector of p-2 indices, cb_c in (13) is a matrix with p-2 columns.

Finally, sub-facets are defined by vertices obtained by summing the displacement vector, the offset, and the four combinations of maximum and minimum values of the two free controls, u_a and u_b. For instance, the four vertices for the q^{th} max sub-facet are

$$\begin{aligned} m_{1,q} &= m_{\max} + cb_a u_{a,\min} + cb_b u_{b,\min} + offset_q \\ m_{2,q} &= m_{\max} + cb_a u_{a,\min} + cb_b u_{b,\max} + offset_q \\ m_{3,q} &= m_{\max} + cb_a u_{a,\max} + cb_b u_{b,\min} + offset_q \\ m_{4,q} &= m_{\max} + cb_a u_{a,\max} + cb_b u_{b,\max} + offset_q \end{aligned} \tag{14}$$

3.8. Computation of the Control Input

With all but two of the controls at their limits for a given sub-facet, the control that will achieve a moment vector intersecting the sub-facet can be defined as before. However, since overlapping sub-facets may exist at a particular boundary point, there is not, in general, a unique relationship between the moment and the value of the input needed to achieve it. While the solution corresponding to any sub-facet could be taken as a solution to the direct allocation problem, we propose to take instead the average of the inputs resulting from all overlapping sub-facets. Taking the average is a simple solution that gives the desired moment, and usually reduces the number of saturated controls, as well as guarantees the continuity of the solution.

3.9. Sequential Search for Control Allocation

The sequential search procedure described in section 3.4 may be employed for the search for the right sub-facet. Although all sub-facets containing the desired moment must be found, once a correct sub-facet is encountered, one only needs to check the other sub-facets lying in the same plane (*i.e.* those with the same parent polygonal facet) to complete the search. The sequential search procedure is useful as a baseline for evaluation.

4. Rapid Search Using Spherical Coordinates: Non-Coplanar Case

4.1. Representation of the AMS in Spherical Coordinates

Because the determination of the applicable facet does not depend on the magnitude of the desired moment, the search may be performed in a 2-dimensional space instead of the original 3-dimensional space. Each vertex of the moment space, determined by (x, y, z) coordinates, can be expressed in spherical coordinates (θ, $s\varphi$, ρ), with

$$\theta = \tan^{-1}(y\ ,\ x) \tag{15}$$

$$s\varphi = \sin(\varphi) = \frac{z}{\rho} \tag{16}$$

$$\rho = \sqrt{x^2 + y^2 + z^2} \tag{17}$$

θ represents the *azimuth angle* (the horizontal angle in the x, y plane), φ represents the *elevation angle* (the vertical angle from the x, y plane), and ρ represents the distance from the origin. This third spherical coordinate is irrelevant for the search of the facet. For the azimuth, note that a two-argument inverse tangent function is used. The value $\sin(\varphi)$ (henceforth abbreviated $s\varphi$) is also used instead of φ to simplify on-line computations.

Figure 5 shows the result of transforming the boundary of the C-17 AMS to spherical coordinates, with θ shown on the x-axis in the range of $\pm\pi$. The sine of the elevation angle is shown on the y-axis. The figure shows that the lines that form the edges of the facets become curves, because of the nonlinear change of coordinates. In fact, these curves are the well-known *great circles* used in navigation. They are the projection on the unit sphere of 3D line segments, or the intersection of the unit sphere with a plane including the origin and the two vertices. The idea of using the spherical coordinates is that the desired moment is represented in the 2D space as a point, and that the control allocation problem becomes the simpler problem of determining to which 2D facet the point belongs.

Some terminology is introduced to help convey the algorithms. An *edge* joins two adjacent vertices of a facet. A *crossing* is defined as an edge between two vertices, with azimuth angles θ_1

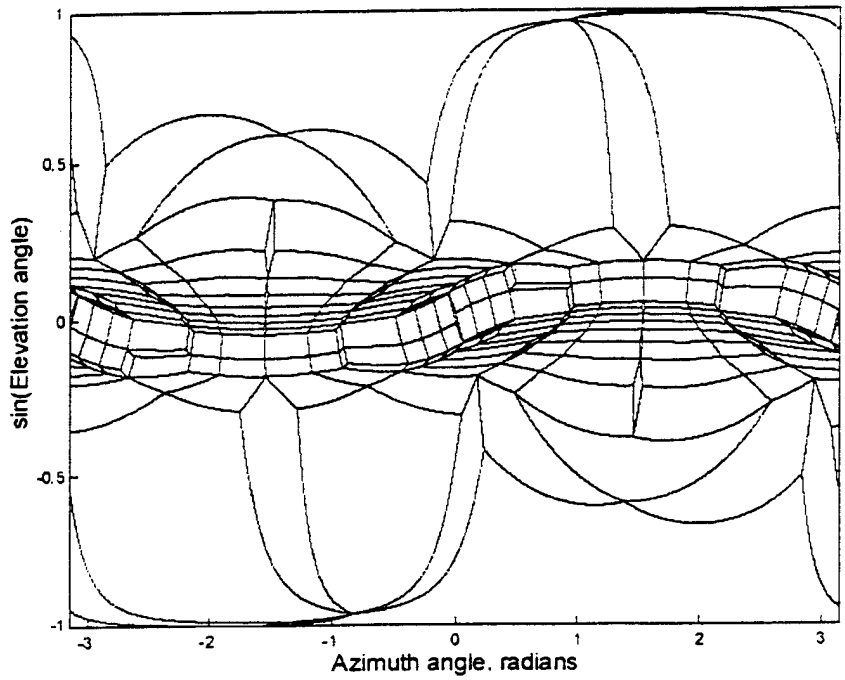


Figure 5. Spherical mapping of C-17 AMS facets.

and θ_2, which crosses the $\pm\pi$ boundary. It is found by testing

$$\delta\theta = \theta_{max} - \theta_{min}, \tag{18}$$

with $\theta_{max} = \max(\theta_1, \theta_2)$, $\theta_{min} = \min(\theta_1,\theta_2)$. If $\delta\theta > \pi$, one may conclude that the edge crosses the boundary. A *split facet* is one in which the facet is bisected by the line $\theta = \pm\pi$.

The great circle for each pair of vertices looks like a distorted sinusoid in the spherical coordinate space. The curve reaches maximum and minimum values of the elevation angle φ that have equal magnitude and opposite sign. These points occur 180° apart in azimuth angle. For the mapping of two vertices in 3D to two vertices in spherical coordinate space $\{(x_1, y_1, z_1), (x_2, y_2, z_2)\} \rightarrow \{(\theta_1, s\varphi_1), (\theta_2, s\varphi_2)\}$ it turns out that one of the extrema of the great circle occurs at (θ_{pk} , $s\varphi_{pk}$) given by

$$\theta_{pk} = \tan^{-1}(x_1z_2 - x_2z_1,\ y_2z_1 - y_1z_2) \tag{19}$$

$$s\varphi_{pk} = \frac{s\varphi_1}{\sqrt{s\varphi_1^{\,2} + (1 - s\varphi_1^{\,2})(\cos(\theta_{pk} - \theta_1))^2}} \tag{20}$$

To obtain the correct values, θ_{pk} needs to be adjusted by $\pm\pi$ using the following rule:

$$\begin{aligned} &if \quad \cos\left(\theta_{pk}-\theta_{l}\right)<0 \\ &then \ \ \theta_{pk}=\theta_{pk}-sign\left(\theta_{pk}\right)\pi \end{aligned} \tag{21}$$

The other extremum of the great circle is obtained by symmetry.

With the knowledge of the peaks of the great circle, the equations defining the great circle, that is, the edge of the facet under consideration, is given by

$$\begin{aligned} &\cos\left(\varphi_{pk}\right)=1-s\varphi_{pk}^{2} \\ &\alpha=s\varphi_{pk}\cos\left(\theta_{pk}-\theta_{k}\right) \\ &s\varphi_{k}=\frac{\alpha}{\sqrt{\cos(\varphi_{pk})+\alpha^{2}}} \end{aligned} \tag{22}$$

where $(\theta_k, s\varphi_k)$ is a point on the great circle.

4.2. Rapid Search using Facet Boxes

In the first option, ranges are computed for the coordinates of the facets, and they define boxes in which the facets are located. The boxes are used to quickly assess whether the desired moment is likely to lie within a given facet. The overall approach is similar to the exhaustive search, but simple inequality tests are used to drop facets from the list. For those facets that are left, the usual 3D test is performed. If the test is successful, the control input is quickly obtained. Otherwise, the search continues. The idea is that the 3D test is then required for very few facets.

4.2.1. Off-line Computations

Off-line computations consist in the determination of the AMS, and of the boxes that delimit the facets in spherical coordinates. For illustration, a facet box is outlined with a dashed line in figure 6. It shows that the box is determined not only by the coordinates of the vertices, but also by maxima reached within the edges of a facet. The peaks of the great circles are therefore determined using (19), (20), and (21), and their values are used in the computations of the box if the peaks lie between the vertices.

A difficulty with the implementation of the method is that facets may span the boundaries of the 2D space. In particular, two facets include the *north pole* and the *south pole*. The north and

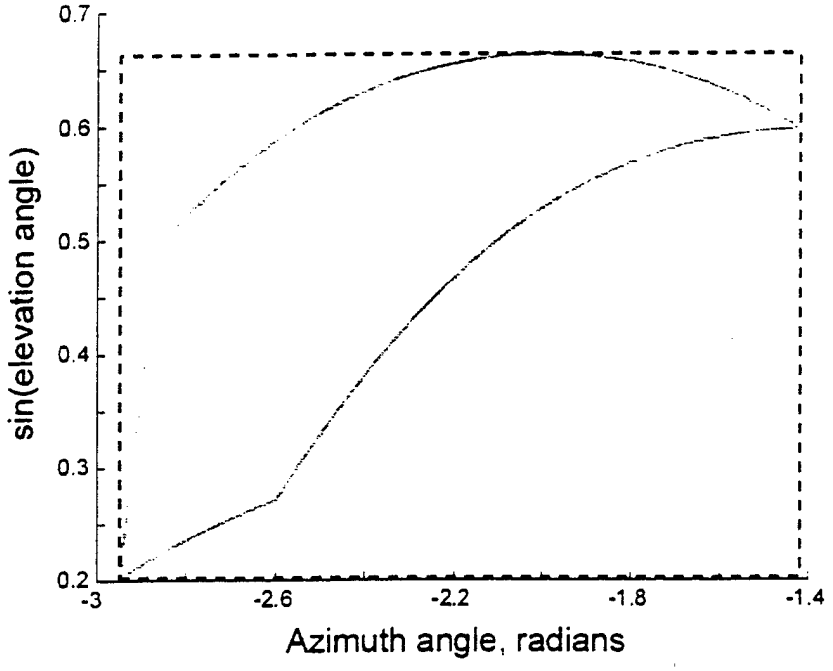


Figure 6. Spherical mapping of a single facet outlined by a facet box.

south poles are the points with $\varphi = 90^\circ$ and $\varphi = -90^\circ$, respectively. Facets may also span the azimuth boundaries. Such facets could be split into two facets to perform box tests. Instead, however, facet types are defined, and those facets that span the azimuth boundary are redefined on the $(0, 2\pi)$ range so that they become contiguous. The procedure is then implemented as follows.

Step 1: *Compute AMS vertex coordinates and facets.* The AMS is computed using the direct method as described above.

Step 2: *Compute spherical coordinates of all vertices.*

Step 3: *Determine the facet type.* Five types of facets are considered and are designated as types 0, 1, 2, 3, and 4. The azimuth range is extended so that each facet is completely contained in at least one of the two ranges of θ: $-\pi < \theta < \pi$ and $0 < \theta < 2\pi$. A facet is assigned a label of type 0 (those facets in the first set) or type 2 (those facets in the second set, *i.e.* split facets). Further, three special cases are assigned: (a) facets that enclose a pole (type 1), (b) facets that border a pole and lie within $-\pi < \theta < \pi$ (type 3), and (c) facets that border a pole and lie within the range $0 < \theta < 2\pi$ (type 4). Facet type can be determined by testing the $\delta\theta$ of each facet edge. The

value of $\delta\theta$ is categorized in four possible ranges:

$$\textit{Case 1: } \delta\theta < \pi, \quad \textit{Case 2: } \delta\theta > \pi, \quad \textit{Case 3: } \delta\theta = \pi, \quad \textit{Case 4: } \delta\theta = 0.$$

A simple algorithm can be applied to determine the facet type based on the type and number of crossings, with

$$\textit{facet type} = c + 3d \tag{23}$$

where $c \in \{0,1,2\}$ is the number of crossings and $d \in \{0,1\}$ is the number of occurrences of case 3.

Step 4: *Determine box boundaries of spherical facets.*

(a) *Compute the maximum and minimum spherical coordinates of the edges between vertices of each facet.* If the facet is type 2 or type 4 (*i.e.*, in the $0 < \theta < 2\pi$ range), negative values of θ_1, θ_2, and θ_{pk} of each edge are incremented by 2π before calculating $s\varphi_{pk}$.

(b) *Store the extremal values of* θ *and* $\sin(\varphi)$ *for each facet to define the facet box.* For each facet, determine θ_{min}, θ_{max}, $s\varphi_{min}$, and $s\varphi_{max}$. Store these values in a facet box table. If the facet includes a north (south) pole, $s\varphi_{max}$ ($s\varphi_{min}$) is forced to its maximum (minimum) of 1 (-1). Distinguishing a north pole from a south pole can be done by calculating the great circle formed by any two of the facet vertices that form an edge and testing the location of a third facet vertex (θ_3, $s\varphi_3$) relative to this great circle. If $s\varphi_{gc}$ is computed from (22) with $\theta_k = \theta_3$, and if $s\varphi_3 > s\varphi_{gc}$, the facet is a north pole. Otherwise, it is a south pole.

4.2.2. On-line Computations

Step 1: *Convert the desired moment into spherical coordinates.*

Step 2: *For each facet, check the feasibility of the desired moment.* Compare the coordinate of the desired moment to the facet box. If the facet box is type 2 or type 4, add 2π to the azimuth of the point. If the desired moment lies within the box boundaries, compute the 3x3 inverse of section 3.3. The rest of the controls are given by the control flags associated with the facet number.

Step 3: *Compute the control input.* Once the correct facet is found and the 3D test is performed, one only needs to scale the control as necessary to satisfy the constraints.

4.3. Rapid Search using Table Look-up

The second option consists in creating a look-up table $f(\theta, s\varphi)$ which gives the number of the facet associated with a given pair of spherical coordinates. If the azimuth and elevation angles are quantized with 1000 points each, this option requires an array with 1,000,000 values, a size which is large but within the reach of existing computers. The creation of the table is essentially the transcription of figure 5 into an array, and the marking of the elements of the array with the associated facet number.

The on-line computations could not be simpler with this approach: the facet towards which the desired moment points is found instantly by table look-up, and the appropriate control is determined with minor computations. Note that control allocation is guaranteed to be performed within a known and short period of time.

4.3.1. Off-line Computations: Construction of the Facet Table

The steps to this method are as follows.

Step 1: *Compute AMS vertex coordinates and facets.* The AMS is computed using the direct method as before. From this computation, one obtains coordinate information as well as knowledge of which vertices connect along an edge of the facet, and the set of controls that form each facet.

Step 2: *Compute spherical coordinates of the vertices.* For a vertex at (x, y, z), the spherical coordinates $(\theta, s\varphi, \rho)$ are obtained using eqs. (15, 16, and 17).

Step 3: *Compute and quantize the facet edges.* Quantizing the edges is done by converting the end points to a range of index values in θ, computing corresponding values of $s\varphi$, and then converting these values to an appropriate index. Continuous edges in the range $-\pi<\theta<\pi$ are straightforward. Edges which cross the $\theta = \pm\pi$ boundary have $s\varphi$ indices that are not contiguous and must be managed properly. The indices of each edge of the facet are stored in a common array.

Step 4: *Modify pole facets.* Facets containing a pole must be treated as special cases. A facet

with one crossing, that is, one edge with $\delta\theta = \pi$ (corresponding to an edge that passes through a pole), contains a pole. Since the line $s\varphi$ is actually a single point in 3D, the edge $s\varphi = \pm 1$ (1 for north pole, -1 for south pole) for each θ index must be added to the common array of step 3.

Step 5: *Create facet table.* The facet table is a two-dimensional array of the facet numbers or identifiers. The indices represent quantized values of θ and $s\varphi$. When the common array is complete, the facet table is updated in the storage locations to which the facet indices just calculated correspond, including indices inside the boundary of the facet.

4.3.2. On-line Computations

Step 1: *Convert the desired moment into spherical coordinates and then to facet table indices.*

Step 2: *Obtain the facet number from the look-up table.*

Step 3: *Compute the control input.* Compute the 3x3 inverse of section 3.3 to arrive at the values for the free controls. The rest of the controls are given by the control flags associated with the facet number. Scale the control as necessary to satisfy the constraints.

4.4. C-17 Example

Each algorithm was tested with 1000 randomly selected desired moments for the C-17 example. The AMS has 240 facets and 242 vertices in moment space. The sequential search method was used to establish a baseline to evaluate the other search methods. The spherical facet table technique was simulated using quantizations of 100 and 1000 for each axis.

Figures 7 and 8 display comparisons of the number of floating point operations (obtained by using the *flops* command in MATLAB®) of the three algorithms. These histograms are intended only to provide a rough comparison of the algorithms described in this paper and should be used with caution, as the results are dependent on the specific implementation as well as on the language and hardware used. Note in particular that MATLAB® counts as floating pointing

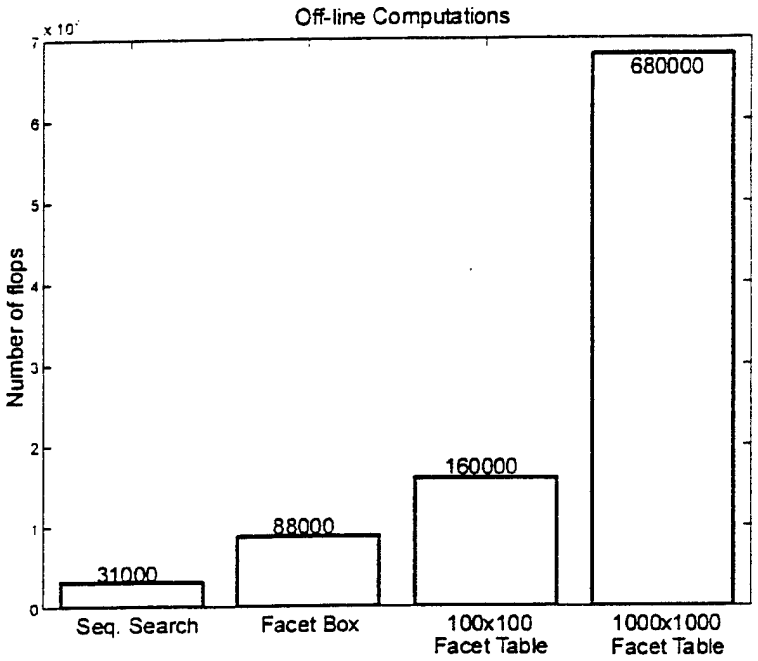


Figure 7: Histogram showing comparison of off-line computations for the different methods.

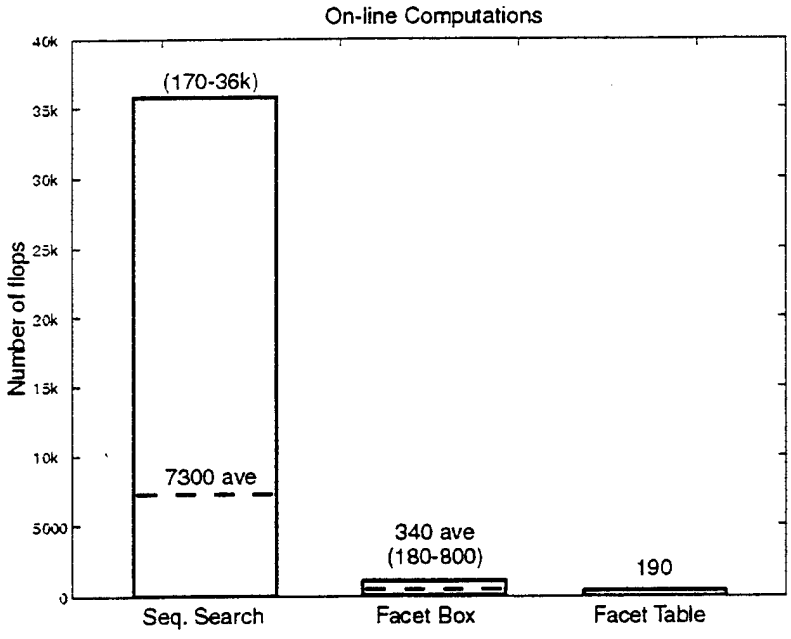


Figure 8: Histogram showing comparison of on-line computations for the different methods.

operations some operations that would normally be counted as integer operations. The code for each algorithm was written in MATLAB® version 5.3.

The histogram in figure 8 shows a range and an average value for the on-line code of the sequential search and facet box methods. The range denotes the minimum to maximum possible values for the method, whereas the average was computed for the 1000 randomly sampled moments. One finds that the facet table approach considerably reduces the number of required computations to be performed on-line, and eliminates the variability in the number of those computations. Memory requirements are significant, however. In an adaptive control application, the off-line computations may also constitute an important burden to be considered. The facet box approach is a simple and useful intermediate option.

Further analysis indicates some interesting characteristics of the facet box algorithm. Figure 9 gives a histogram of the number of box tests performed before the correct facet is found. The theoretical maximum is 240 in this example, but one finds that rarely more than 150 tests are required. The average number of tests was computed to be 49.7.

With the box test, the number of 3x3 inverses to be performed is considerably less than the number of facets tested. Figure 10 shows that in nearly 50% of the cases, only one facet was tested. In 95% of the cases, a maximum of three inverses was required, and in none of the cases were more than five inverses required. The average number of 3D tests was computed to be 2.0, to be compared with 49.7 required without the box test. While sequential search by itself is impractical, box tests in spherical coordinates make this approach much more feasible. The number of computations required, however, is not fixed.

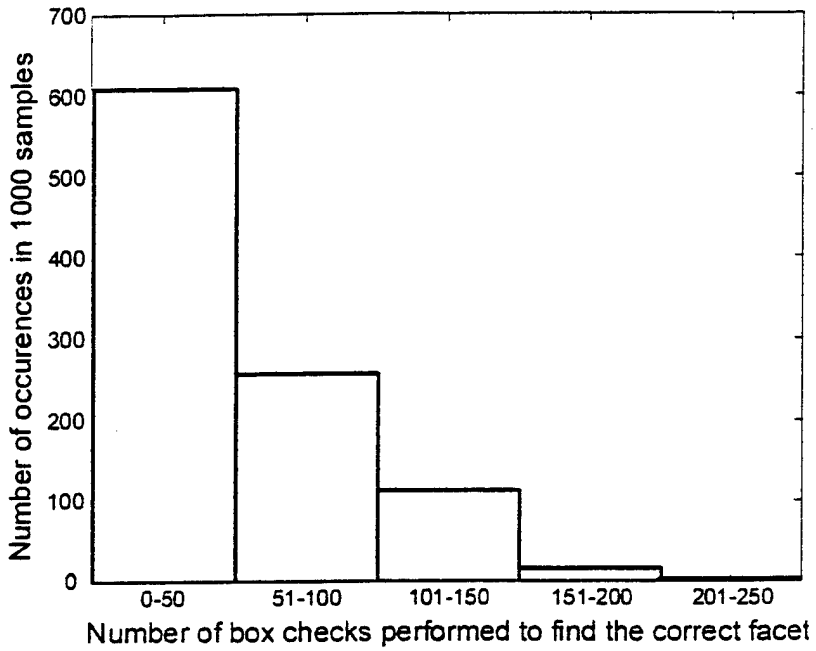


Figure 9: Histogram of facets checked.

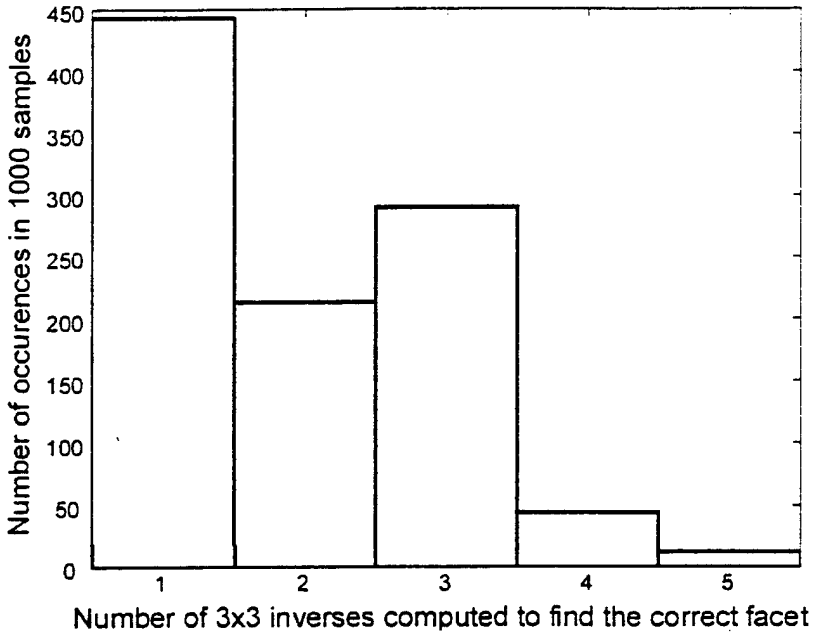


Figure 10: Histogram of the number of 3x3 inverses computed.

5. Rapid Search Using Spherical Coordinates: Coplanar Case

5.1. Representation of the AMS in Spherical Coordinates: Coplanar Case

Figure 11 shows the result of transforming the boundary of the advanced tailless fighter AMS to spherical coordinates. The transformed polygonal facets are visible. Although not shown, parallelogram sub-facets are used to cover the area of every polygonal facet. Because the polygonal facets can be completely specified by sub-facets, the same options as discussed in sections 4.2 and 4.3 can be used. However, modifications to the specific options are made here to extend its use to solve the problem of overlapping sub-facets.

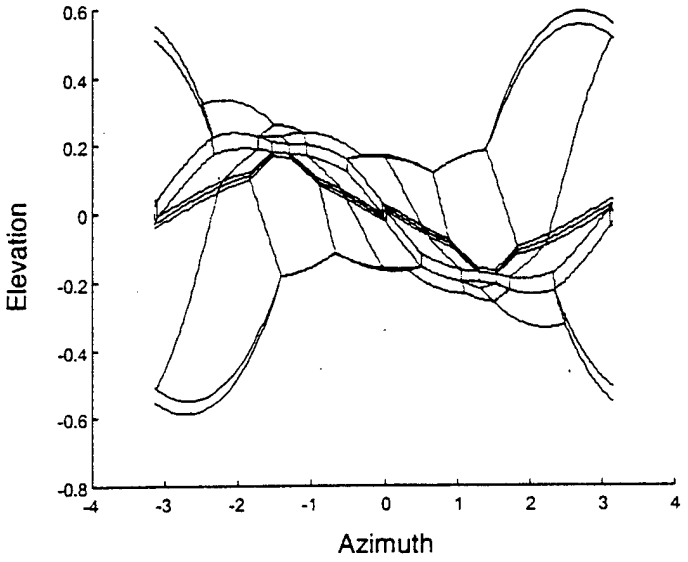


Figure 11. Spherical mapping of overlapping facets outlined by facet boxes.

5.2. Rapid Search using Facet Boxes

5.2.1. Off-line Computations

As in the original method, ranges are computed for the coordinates of the sub-facets. Sub-facets are treated no different from non-coplanar facets. However, a methodology for tracking overlapping sub-facets improves the on-line search. An array identifies a parent polygonal facet for each sub-facet. Once a sub-facet is found, only the other sub-facets associated with the same parent are tested. For our example of the tailless fighter, which has 78 polygonal facets and 210

sub-facets, the array has 210 rows and 78 identifiers. Figure 12 shows an illustration of overlapping sub-facets outlined with facet boxes.

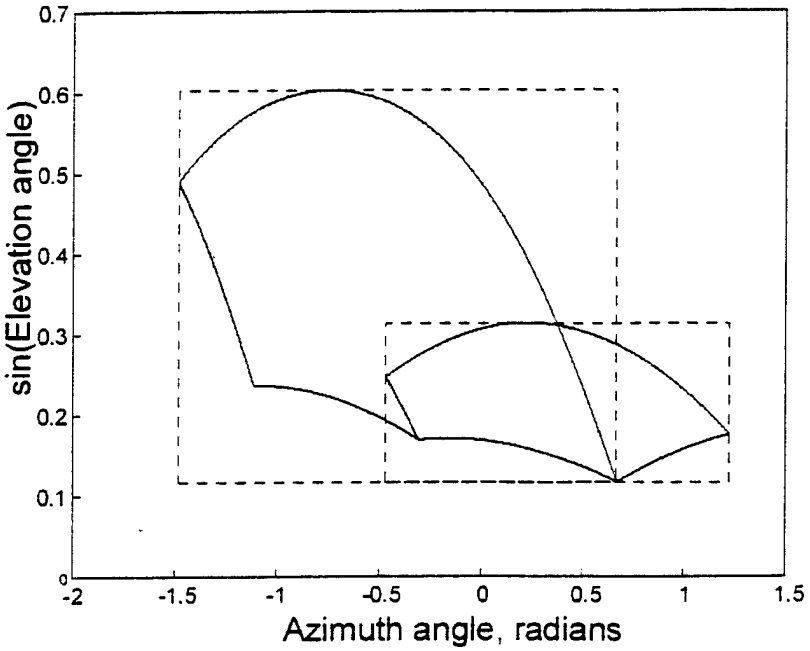


Figure 12: Spherical mapping of overlapping sub-facets outlined by facet boxes.

5.2.2. On-line Computations

Step 1: *Convert the desired moment into spherical coordinates.*

Step 2: *For each sub-facet, check the feasibility of the desired moment.* Compare the coordinate of the desired moment to the facet box. If the desired moment lies within the box boundaries, compute the control for this sub-facet. If the conditions on (ρ_1, ρ_2, ρ_3) are satisfied, check all other sub-facets associated with the same parent facet. If not, go to the next sub-facet.

Step 3: *Compute the control input.* Once the correct sub-facets are found and the 3D tests are performed, the results are averaged and scaled if the desired moment exceeds the achievable value.

5.3. Rapid Search using Table Look-up

The creation of a spherical coordinate table for a system with coplanar controls can be done by including a third dimension. The added dimension is a vector of n_{pf} values corresponding to the sub-facets that overlap at that location. Low quantization resolution may increase the apparent number of overlapping sub-facets.

5.3.1. Off-line Computations: Construction of the Facet Table

The facet table is a two-dimensional array of sub-facet numbers or identifiers. Sub-facet indices are recorded along a third dimension, if overlapping occurs. Again, the details of this method are identical to those explained in section 4.3 with the modification of the vectorized facet identifiers in the table.

5.3.2. On-line Computations

Step 1: *Convert the desired moment into spherical coordinates and then to indices of the facet table.*

Step 2: *Obtain the array of facet numbers from the look-up table.*

Step 3: *Compute the control input.* Compute the 3x3 inverse of section 3.3 for each sub-facet in the array to arrive at the values for the *free* controls. Average the resulting controls of each sub-facet. Scale the averaged control as necessary.

Again, the on-line computations are quite simple with the facet table approach: the sub-facets towards which the desired moment points are found instantly by table look-up, and the appropriate control is determined with minor computations. Note that on-line computations could be even further reduced by converting the facet table into a *control* table. This would be done by computing the control for each facet table location in the off-line code. The on-line code then becomes exclusively a look-up table to obtain the control. This method is ideal for applications where low resolution is acceptable or where simple and reliable on-line code is of premium importance.

5.4. Tailless Fighter Example

Each algorithm was tested with 1000 randomly selected desired moments and with the tailless fighter model. The AMS has 210 sub-facets in moment space. The sequential search method was used to establish a baseline to evaluate the other search methods. The spherical facet table technique was implemented using quantizations of 100 and 1000 for each axis. Figure 13 displays a comparison of floating point operations for the off-line computations. The number of computations for the 1000x1000 facet table was of the order of 10^7 flops and was not plotted with the others in the histograms.

Figure 14 shows the on- line computations. The histogram entries for the sequential search and facet box methods have a range and an average value. The range denotes the minimum to maximum possible values for the method, whereas the average was computed for the 1000 randomly sampled moments. As in the case for systems with non-coplanar controls, both spherical approaches considerably reduce the number of computations to be performed on-line, and significantly reduces the variability in the number of those computations. The facet box approach is nearly equivalent to the facet table method in terms of on-line calculations and significantly cheaper in terms of off-line computations.

The variation in on-line computations for the facet table method of different quantizations is due to variations in the number of apparent overlapping sub-facets. Although two sub-facets might not overlap, if the quantization is low enough, they effectively may. This results in more sub-facets to be checked and averaged. Therefore a higher quantization will typically yield lower on-line computations.

Although the theoretical maximum number of box tests is 210 the average number is only 52. Figure 15 gives a histogram of the number of box tests performed before the correct facet is found. Figure 16 shows a histogram of the number of 3x3 inverses computed before a final control value is determined. In 92% of the cases, a maximum of eight inverses was required, and in none of the cases were more than fifteen inverses required. The average number of 3D tests was computed to be 5.6, to be compared with 52 required without the box test. The number of computations required, however, is not fixed due to variability in the number of overlapping facets.

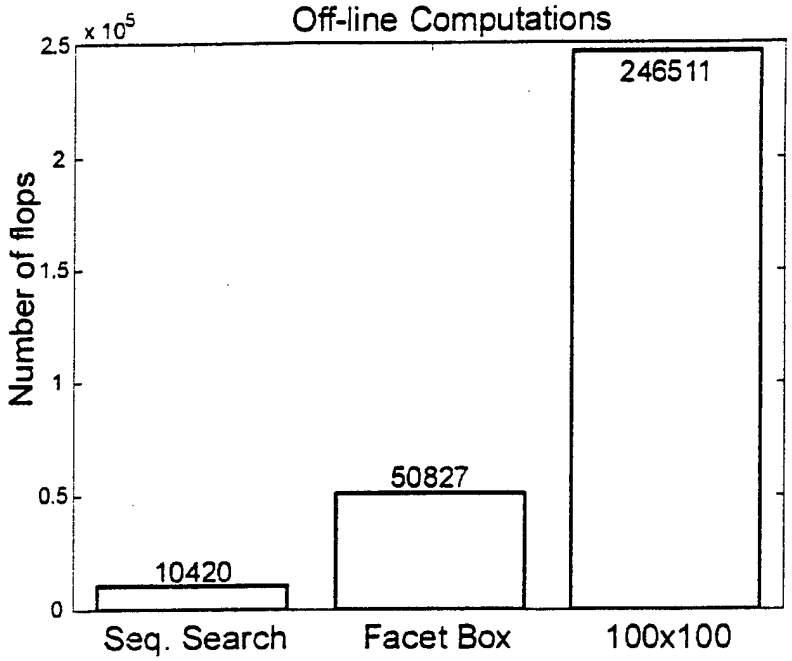


Figure 13. Histogram showing comparison of off-line computations for direct allocation methods for an 11 actuator coplanar system.

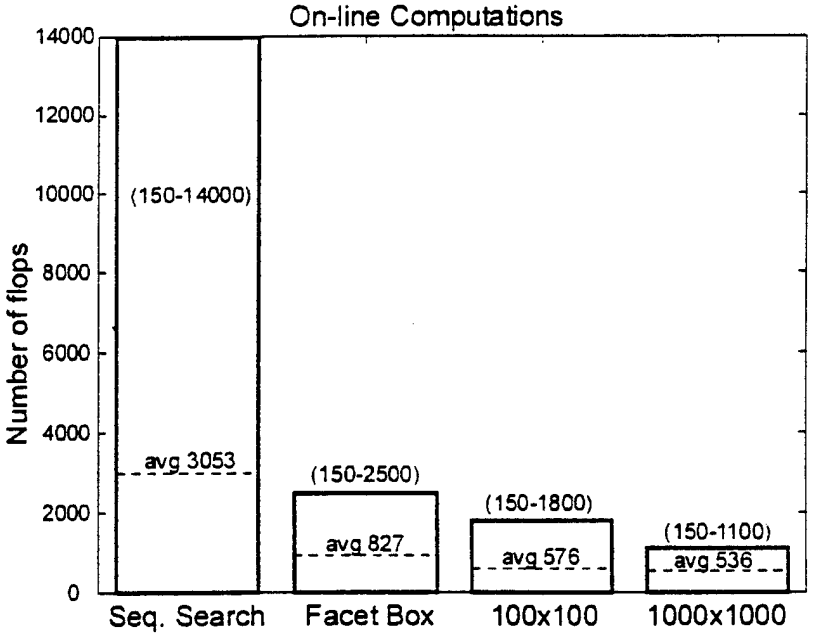


Figure 14. Histogram showing comparison of on-line computations for direct allocation methods for an 11 actuator coplanar system.

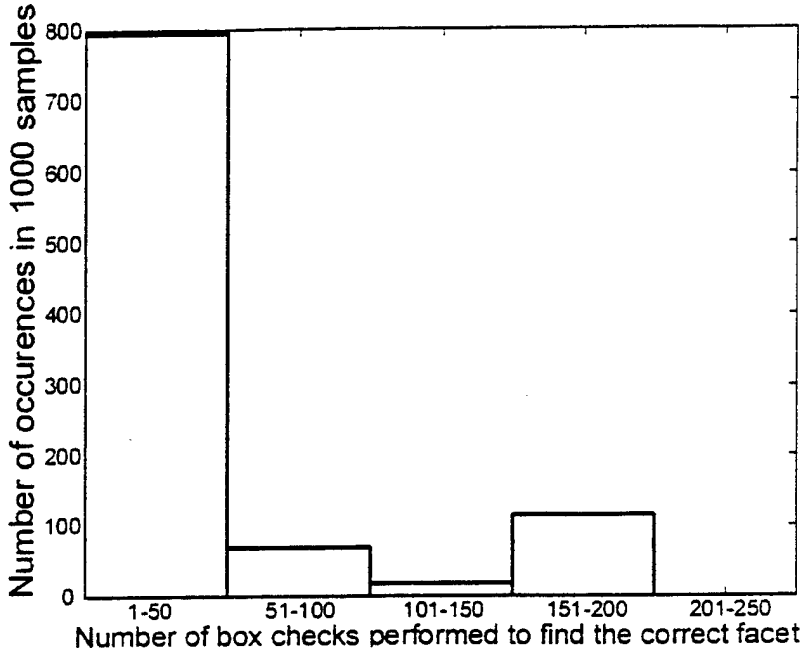


Figure 15. Histogram of sub-facets checked.

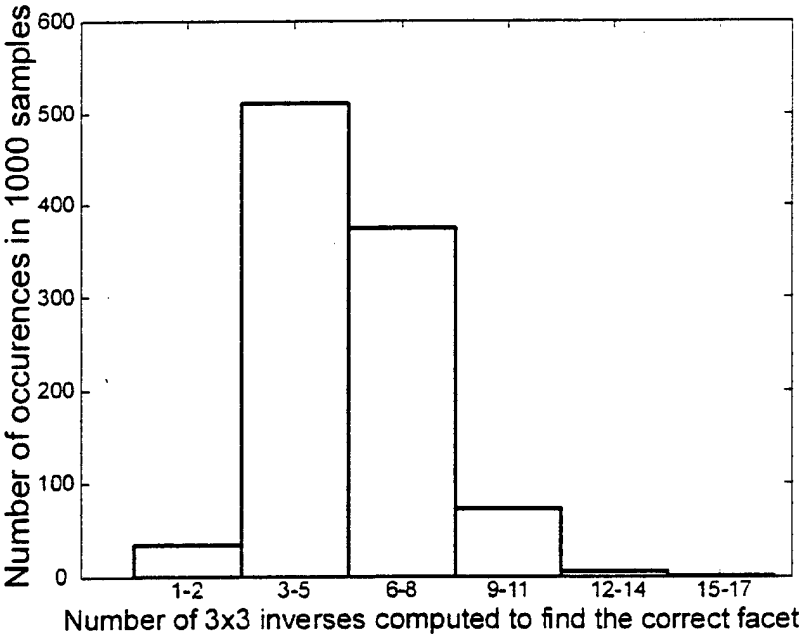


Figure 16. Histogram of the number of 3x3 inverses computed.

6. Conclusions

Direct allocation provides a solution to the control allocation problem that not only retains the direction of the desired moment, but also yields a solution that takes advantage of the maximum attainable moment set. Direct allocation was previously only applicable to systems whose controls effectiveness matrix was such that every three columns were linearly independent. A system that is not limited in this way has been called a system with coplanar controls, or simply a coplanar system. The example of a tailless fighter aircraft model was shown to fall in this category.

The geometry describing the attainable moment set for a coplanar system was explained and an extension to the direct allocation method was given in this paper. The average of the multiple solutions was computed in the procedure, and the concept of overlapping sub-facets was found useful for that purpose.

The representation of the AMS in spherical coordinates makes it possible to rapidly perform the on-line computations required by the direct allocation method. No prior information is required about the approximate location of the correct facet. Two options were discussed which have their respective advantages. The first option (facet box method) did not require large memory storage, but had a larger and variable number of on-line computations. The number of computations for a given control cycle will not exceed $n(n\text{-}1)$ box check comparisons (trivial) and a few 3x3 inverses. The number of 3x3 inverses is uncertain but was found to not exceed 5 in our tests involving an aircraft model with 16 actuators.

The second option (facet table look-up method) required virtually no on-line computations and provided a guaranteed solution in a fixed time. The drawback was a potentially large memory requirement and longer off-line execution time. Overall, both options provide a considerable improvement over a sequential search of the facets based on the 3D test.

Slight modifications to the spherical coordinate methods were shown to provide direct allocation solutions to coplanar systems. The properties of those methods were similar to those for non-coplanar systems. Overlapping sub-facets were identified by adding a third dimension to the table that stores the identifier of each sub-facet.

The facet table approach can be viewed as a non-linear extension of standard ganging techniques. Table look-up replaces conventional linear transformations. The advantage over

ganging and other simple control allocation techniques is that it guarantees the use of the maximum control authority available, while requiring very few computations. Compared to other rapid search techniques for direct allocation, its advantage is a high degree of predictability and reliability. However, the method requires a significant amount of memory and is not well suited to reconfigurable control, which would require continuous update of the look-up table in real-time.

7. References

[1] J. Brinker & K. Wise, "Reconfigurable Flight Control for a Tailless Advanced Fighter Aircraft," *Proc. of the AIAA Guidance, Navigation, and Control Conference*, Paper AIAA 98-4107, Boston, MA, August 1998.

[2] R. Eberhardt & D. Ward, "Indirect Adaptive Flight Control of a Tailless Fighter Aircraft," *Proc. of the AIAA Guidance, Navigation, and Control Conference*, Paper AIAA 99-4042, Portland, OR, August 1999.

[3] D. Enns, "Control Allocation Approaches," *Proc. of the AIAA Guidance, Navigation, and Control Conference*, Paper AIAA 98-4109, Boston, MA, August 1998.

[4] W. C. Durham, "Constrained Control Allocation," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 4, 1993, pp. 717-725.

[5] W. C. Durham, "Constrained Control Allocation: Three-Moment Problem," *Journal of Guidance, Control, and Dynamics*, Vol. 17, No. 2, 1994, pp. 330-336.

[6] W. C. Durham, "Attainable Moments for the Constrained Control Allocation Problem," *Journal of Guidance, Control, and Dynamics*, Vol. 17, No. 6, 1994, pp. 1371-1373.

[7] K. A. Bordignon & W. C. Durham, "Closed-Form Solutions to Constrained Control Allocation Problem," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 5, 1995, pp. 1000-1007.

[8] J. Buffington & P. Chandler & M. Pachter, "Integration of On-line System Identification and Optimization-based Control Allocation," *Proc. of the AIAA Guidance, Navigation, and Control Conference*, Paper AIAA 98-4487, Boston, MA, August 1998.

[9] M. Bodson & W. Pohlchuck, "Command Limiting in Reconfigurable Flight Control," *Journal of Guidance, Control, and Dynamics*, Vol. 21, No. 4, July-August 1998.

[10] W. C. Durham, "Computationally Efficient Control Allocation," *Proc. of the AIAA Guidance, Navigation, and Control Conference*, Paper AIAA 99-4214, Portland, OR, August 1999.

[11] J. Buffington, "Tailless Aircraft Control Allocation," *Proc. of the AIAA Guidance, Navigation, and Control Conference*, Paper AIAA 97-3605, New Orleans, LA, August 1997.